

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

FRANCIS VAGNER DOS ANJOS FONTOURA
GABRIEL DIAS SCHMOELLER

BICHO PERDIDO
UM SISTEMA PARA RECUPERAÇÃO DE ANIMAIS DE ESTIMAÇÃO PERDIDOS
BASEADO NA WEB E EM PLATAFORMAS MÓVEIS

Porto Alegre
2015

FRANCIS VAGNER DOS ANJOS FONTOURA
GABRIEL DIAS SCHMOELLER

BICHO PERDIDO
UM SISTEMA PARA RECUPERAÇÃO DE ANIMAIS DE ESTIMAÇÃO PERDIDOS
BASEADO NA *WEB* E EM PLATAFORMAS MÓVEIS

Trabalho de conclusão de curso de graduação apresentado à Faculdade de Informática da Pontifícia Universidade Católica do Rio Grande do Sul como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Profa. Dra. Cristina Moreira Nunes

Porto Alegre
2015

RESUMO

Animais de estimação, mais do que ótimas companhias, são como parte da família. Apesar do carinho e cuidado dos donos, cerca de um terço destes animais, em especial cães e gatos, se perdem pelo menos uma vez durante sua vida e apenas 10% são encontrados. Acreditando no potencial da Tecnologia da Informação para mitigar esse problema, foi construído o Bicho Perdido, um sistema para a *web* e plataformas móveis, que permite o cadastro e a busca de anúncios de animais perdidos e encontrados. Além disso, esse sistema atua como um Sistema de Recomendação, comparando os anúncios cadastrados, calculando suas semelhanças e auxiliando o usuário, exibindo sugestões de anúncios relevantes à sua procura. O Bicho Perdido cumpre seu objetivo, utilizando as mais recentes tecnologias abertas da *web*, as capacidades dos dispositivos móveis, a tecnologia Java e o conceito de sistemas distribuídos (*web services*). Porém, dada a evolução constante da computação, é reconhecido o amplo espaço para a sua melhoria, seja em questão de capacidades ou desempenho, para que cada vez mais animais perdidos possam retornar aos seus lares, saudáveis e o mais rapidamente possível. Este documento detalha as funcionalidades do sistema, o algoritmo de recomendação, a arquitetura do *software* e a estratégia de desenvolvimento e integração.

Palavras-chave: Animais Perdidos. Sistemas para a *Web*. Aplicativos Móveis. Sistemas de Recomendação.

ABSTRACT

Pets are like family. About one third of dogs and cats are lost at least once during their lifetime and only 10% are found. We believe in the Information Technology potential to mitigate this problem and we built the *Bicho Perdido* (Lost Pet) system. It is made for web and mobile platforms and allows publication and search for lost and found pets advertisements. In addition, this system acts as a Recommendation System, comparing the registered ads, calculating their similarities and helping the user by displaying suggestions of relevant ads to your search. The *Bicho Perdido* fulfills its purpose, using the latest open web technologies, mobile device capabilities, the Java technology and the distributed systems concept (web services). However, given the constant computing evolution, there is room for its improvement, in capacity or performance, so lost pets can return to their homes as soon as possible. This document details the system features, the recommendation algorithm, the software architecture and the development and integration strategy.

Keywords: Lost Pets. Systems for the Web. Mobile Applications. Recommendation Systems.

LISTA DE ILUSTRAÇÕES

Figura 1 – População de animais no Brasil.	11
Figura 2 – Ortodromia entre os pontos A e B.	16
Figura 3 – Abordagens aditiva (RGB) (esquerda) e subtrativa (CMYK) (direita).	21
Figura 4 – Busca de anúncios no Cachorro Perdido.	23
Figura 5 – Lista de relatos no Canil Virtual.	24
Figura 6 – Cadastro para recebimento de alertas no Canil Virtual.	25
Figura 7 – Ocorrência em detalhes no Procura-se Cachorro.	26
Figura 8 – Busca no Procura-se Cachorro.	27
Figura 9 – Telas do aplicativo móvel Procura-se Cachorro.	27
Figura 10 – Telas de criação e autenticação de usuários.	29
Figura 11 – Telas de criação de um anúncio.	31
Figura 12 – Esboço de seleção de cores pré-definidas.	32
Figura 13 – Telas de definições de usuário protetor.	32
Figura 14 – Telas de busca por anúncios.	33
Figura 15 – Telas de geração de cartazes.	33
Figura 16 – Telas de sugestões de anúncios semelhantes.	34
Figura 17 – Exemplo de cão com cor duvidosa.	35
Figura 18 – Tala com notificações recebidas.	35
Figura 19 – Exemplo de matriz de semelhança entre cores de um anúncio com 3 cores e outro com 2.	38
Figura 20 – Selecionando o maior grau da matriz.	38
Figura 21 – Removendo a linha e a coluna correspondente ao maior grau.	39
Figura 22 – Repetindo o procedimento para a nova matriz.	39
Figura 23 – Calculando o grau resultante do atributo cor.	39
Figura 24 – Selecionando o maior grau repetido mais próximo da diagonal da matriz.	40
Figura 25 – Corrigindo o grau resultante do atributo cor.	40
Figura 26 – Diagrama de pacotes.	46
Figura 27 – Diagrama de classes do microserviço de cálculo de semelhança.	55
Figura 28 – Gráfico de tempo total de execução do algoritmo por fase e geral.	62
Figura 29 – Gráfico de tempo de execução do algoritmo por comparação no SGBD.	64
Figura 30 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 1000 anúncios.	64
Figura 31 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 2000 anúncios.	65

Figura 32 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 3000 anúncios.	65
Figura 33 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 4000 anúncios.	66
Figura 34 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 5000 anúncios.	66
Figura 35 – Diagrama de casos de uso.	71
Figura 36 – Diagrama de classes detalhando a entidade Usuário.	73
Figura 37 – Diagrama de classes detalhando a entidade Anúncio.	74
Figura 38 – Diagrama de classes de análise de registro, <i>login</i> e protetor.	74
Figura 39 – Diagrama de classes de análise da busca de anúncios.	75
Figura 40 – Diagrama de classes de análise dos detalhes de um anúncio.	75
Figura 41 – Diagrama de classes de análise do cadastro de anúncio.	76
Figura 42 – Diagrama de classes de análise dos próprios anúncios.	76
Figura 43 – Tela de cadastro de usuário.	77
Figura 44 – Tela de autenticação de usuário.	78
Figura 45 – Tela de configuração do usuário protetor.	79
Figura 46 – Opções do menu “Anúncios”	80
Figura 47 – Primeira parte da tela de cadastro de anúncio.	80
Figura 48 – Segunda parte da tela de cadastro de anúncio.	81
Figura 49 – Terceira parte da tela de cadastro de anúncio.	81
Figura 50 – Quarta parte da tela de cadastro de anúncio.	82
Figura 51 – Tela de sugestão de anúncios semelhantes após o cadastro.	82
Figura 52 – Tela de visualização e gerenciamento dos próprios anúncios.	83
Figura 53 – Filtros em comum na tela de busca.	83
Figura 54 – Mapa na tela de busca.	84
Figura 55 – Legenda do mapa na tela de busca.	84
Figura 56 – Resultados de uma busca por palavras-chave.	85
Figura 57 – Parte superior da tela de detalhes de um anúncio.	85
Figura 58 – Parte inferior da tela de detalhes de um anúncio.	86
Figura 59 – Parte superior da tela de previsão do cartaz.	87
Figura 60 – Parte inferior da tela de previsão do cartaz.	87
Figura 61 – Demonstração de um cartaz impresso.	88

LISTA DE TABELAS

Tabela 1 – Estatísticas: quantos dias foi encontrado após o desaparecimento?	12
Tabela 2 – Estatísticas: a que distância se encontrava? (Cães)	12
Tabela 3 – Estatísticas: a que distância se encontrava? (Gatos)	13
Tabela 4 – Comparativo de ferramentas para divulgação de animais perdidos.	36
Tabela 5 – Relação do peso do atributo com grau de relevância.	37
Tabela 6 – Grau de semelhança para o atributo local (distância) entre cães.	41
Tabela 7 – Grau de semelhança para o atributo local (distância) entre gatos.	41
Tabela 8 – Grau de semelhança para o atributo Porte.	41
Tabela 9 – Grau de semelhança para o atributo Pelagem.	42
Tabela 10 – Atributos comparados, seus tipos e pesos.	42
Tabela 11 – Exemplo de atributos exclusivos e normais	43
Tabela 12 – Exemplo de atributos Bônus	44
Tabela 13 – Tabela de códigos de erro do sistema.	56
Tabela 14 – Distribuição da amostra do primeiro teste.	60
Tabela 15 – Distribuição da amostra do segundo teste.	60
Tabela 16 – Distribuição da amostra do terceiro teste.	61
Tabela 17 – Distribuição da amostra do quarto teste.	61
Tabela 18 – Distribuição da amostra do quinto teste.	62
Tabela 19 – Estatísticas coletadas nos testes.	63

LISTA DE ABREVIATURAS E SIGLAS

Abinpet	Associação Brasileira da Indústria de Produtos para Animais de Estimação
API	<i>Application Programming Interface</i>
CCZ	Centro de Controle de Zoonoses
CFMV	Conselho Federal de Medicina Veterinária
CORS	<i>Cross-Origin Resource Sharing</i>
CSS	<i>Cascading Style Sheets</i>
DI	<i>Dependency Injection</i>
ER	Entidade-Relacionamento
EXIF	<i>Exchangeable Image File Format</i>
GUI	<i>Graphical User Interface</i>
GPS	<i>Global Positioning System</i>
HSUS	<i>The Humane Society of the United States</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
HTTPS	<i>Hypertext Transfer Protocol Secure</i>
IDE	<i>Integrated Development Environment</i>
IETF	<i>Internet Engineering Task Force</i>
IoC	<i>Inversion of Control</i>
JDO	<i>Java Data Objects</i>
JFIF	<i>JPEG File Interchange Format</i>
JPA	<i>Java Persistence API</i>
JPEG	<i>Joint Photographic Experts Group</i>
JSON	<i>JavaScript Object Notation</i>

NCPPSP	<i>National Council on Pet Population Study and Policy</i>
ONG	Organizaçāo Não Governamental
ORM	<i>Object-Relational Mapping</i>
REST	<i>Representational State Transfer</i>
SGBD	Sistema de Gerenciamento de Banco de Dados
SMTP	<i>Simple Mail Transfer Protocol</i>
SQL	<i>Structured Query Language</i>
SSE	<i>Server-Sent Events</i>
SSL	<i>Secure Sockets Layer</i>
TCC	Trabalho de Conclusão de Curso
TLS	<i>Transport Layer Security</i>
TI	Tecnologia da Informaçāo
UML	<i>Unified Modeling Language</i>
URI	<i>Uniform Resource Identifier</i>
W3C	<i>World Wide Web Consortium</i>
WHATWG	<i>Web Hypertext Application Technology Working Group</i>
XML	<i>eXtensible Markup Language</i>

SUMÁRIO

Introdução	11
1 REFERENCIAL TEÓRICO	15
1.1 Hypertext Markup Language (HTML)	15
1.2 Geolocalização e Geocodificação	15
1.2.1 Distância entre duas coordenadas geográficas	16
1.3 Sistema de recomendação	17
1.4 Web Service	18
1.4.1 Representational State Transfer (REST)	19
1.4.2 Hypertext Transfer Protocol (HTTP)	19
1.5 JavaScript Object Notation (JSON)	19
1.6 Diferença entre Cores	20
2 TRABALHOS RELACIONADOS	23
3 VISÃO GERAL DO SISTEMA	28
3.1 Motivação e Objetivo Geral	28
3.2 Objetivos Específicos	28
3.3 Funcionamento do Sistema em Detalhes	29
4 ALGORITMO DE RECOMENDAÇÃO	37
4.1 Semelhança por atributo	37
4.2 Semelhança resultante	42
5 IMPLEMENTAÇÃO	45
5.1 Cliente	45
5.1.1 Frameworks utilizados	47
5.1.2 Tipos de aplicativos móveis	47
5.1.3 Módulos	48
5.1.4 <i>Interceptor</i>	49
5.1.5 Local das fotos	50
5.2 Serviço	50
5.3 Integração	54
5.3.1 Tratamento de erros	56
5.3.2 Autenticação	56
5.3.3 Notificações	57

6	TESTE DE DESEMPENHO	59
6.1	População	59
6.2	Testes realizados	60
7	CONCLUSÃO	67
7.1	Trabalhos futuros	67
 REFERÊNCIAS		69
 APÊNDICE A – MODELAGEM		71
 APÊNDICE B – EXEMPLOS DE USO		77
B.1	Cadastrar-se como usuário	77
B.2	Autenticar-se e tornar-se protetor	77
B.3	Criar e gerenciar anúncios	78
B.4	Buscar anúncios	80
B.5	Exibir os detalhes de um anúncio e gerar cartazes	83

INTRODUÇÃO

Os animais de estimação, mais do que ótimas companhias, vêm sendo cada vez mais considerados como membros da família, conforme pesquisas (SUTHERS-MCCABE, 2001). Segundo Costa et al. (2009, p. 4): “Os animais de companhia proporcionam uma significativa melhoria da qualidade de vida das pessoas, aumentando os estados de felicidade, reduzindo os sentimentos de solidão e melhorando as funções físicas e a saúde emocional.” Para Costa et al. (2009 apud SUTHERS-MCCABE, 2001, p. 4): “Os animais de estimação são companhias íntimas que não oferecem competição e podem ser amados sem o medo da rejeição.”

É expressiva a quantidade de mascotes, em especial caninos e felinos, nos lares em todo o mundo. Conforme Wanderson Ferreira, vice-presidente do Conselho Regional de Medicina Veterinária de Goiás na época: “Atualmente, são 21 milhões de gatos no Brasil e 37 milhões de cães. [...] Já nos Estados Unidos [...] são 80 milhões de gatos e 65 milhões de cachorros.” (CFMV, 2014) A Figura 1 apresenta uma visão geral da população de animais no Brasil em 2014.

Figura 1 – População de animais no Brasil.



Fonte /Elaboração: Abinpet
* Dados 2012 - atualização 2º semestre 2014

Fonte: Abinpet

Apesar de todo o carinho e cuidado dos donos, é grande a quantidade de animais desaparecidos. De acordo com dados de Evangelista-Eppenstein (2013 apud NCPPSP; HSUS, ano desconhecido): um terço das mascotes se perdem ao menos uma vez durante sua vida; a cada dois segundos, um animal desaparece e, a cada ano, dez milhões; e apenas 10% são encontrados. O site Encontra-me.org¹, mantido pela Associação Pelos Animais², de Portugal, mostra algumas estatísticas sobre animais perdidos, as quais estão apresentadas nas Tabelas 1, 2 e 3.

Tabela 1 – Estatísticas: quantos dias foi encontrado após o desaparecimento?

Período (dias)	Cães (percentual)	Gatos (percentual)
0 a 1	29,99	25,5
2 a 5	39,98	36,17
6 a 15	19,31	22,93
16 a 30	6,12	8,59
31 a 60	2,75	4,02
61 a 120	1,38	1,18
121 a 180	0,44	0,6
181 ou mais	1,04	1,01

Fonte: Encontra-me.org

Tabela 2 – Estatísticas: a que distância se encontrava? (Cães)

Distância (quilômetros)	Frequencia (percentual)
Menos de 0,5	17,13
0,5 a 1	31,85
2 a 4	28,73
5 a 9	11,22
10 a 14	4,67
15 a 24	3,56
25 a 49	1,6
50 a 99	0,67
100 ou mais	0,58

Fonte: Encontra-me.org

¹ <<https://www.encontra-me.org/>>

² <<http://www.pelosanimais.org.pt/>>

Tabela 3 – Estatísticas: a que distância se encontrava? (Gatos)

Distância (metros)	Frequencia (percentual)
Menos de 150	61,43
151 a 300	13,77
301 a 500	10,91
501 a 1500	7,79
1501 ou mais	6,1

Fonte: Encontra-me.org

As estatísticas mostram que a maioria das mascotes são encontrados nos primeiros dias e próximos ao local do desaparecimento. Então, a ação imediata do dono e a ampla divulgação são essenciais para aumentar as chances de sucesso na busca. Além disso, AgendaPet (2015) recomenda:

- não esperar que o animal volte sozinho;
- divulgar em redes sociais, como Facebook³, Instagram⁴ e Twitter⁵;
 - insistir para que os amigos compartilhem, para maximizar o alcance da informação;
 - detalhar quando e como ocorreu o desaparecimento, se possível, bem como o perfil do bichinho;
 - fornecer fotos e indicar características que facilitem a identificação, como coleira, roupinha, marcas de nascença, etc.;
- espalhar muitos cartazes e panfletos;
 - os cartazes devem ser simples e objetivos;
 - os dizeres “cão (ou gato) perdido” devem estar no topo, em destaque;
 - deve-se usar letras grandes, chamando a atenção dos distraídos e visível de veículos em movimento;
 - deve-se descrever brevemente o animal: gênero, cor predominante, porte, altura do pelo, manchas, raça. Não presumir que a indicação da raça seja suficiente;
 - indicar o nome pelo qual atende: ajuda na captura e apela para o emocional das pessoas;
 - não esquecer das informações de contato;

³ <<http://www.facebook.com>>

⁴ <<http://www.instagram.com>>

⁵ <<http://www.twitter.com>>

- por segurança, não quantificar a recompensa, caso oferecida.
- delimitar um raio de aproximadamente um quilômetro em torno do local da perda e percorrer essa área chamando pelo bicho. Mobilizar a família e amigos para que façam o mesmo. Ampliar o raio de busca a cada tentativa frustrada;
- contatar *pet shops*, clínicas e hospitais veterinários, abrigos e organizações não governamentais (ONG), centros de controle de zoonoses⁶, priorizando os mais próximos aos mais distantes;
- não tentar correr atrás do animal, se encontrá-lo, e deixar que ele venha por si. Dependendo do quanto estará assustado, obter sua confiança não será imediato.

Como foi visto, as recomendações para recuperar um animal perdido consistem em agir imediatamente e divulgar ao máximo. Algumas iniciativas disponíveis na Internet propõem colaborar com a divulgação e a busca, como o Cachorro Perdido⁷, o Canil Virtual⁸ e o Procura-se Cachorro⁹. Suas principais características serão apresentadas no Capítulo 2.

Com base no exposto anteriormente como requisitos, este Trabalho de Conclusão de Curso (TCC) detalha a especificação e construção de um sistema de informação, baseado na *web* e em plataformas móveis, por sua popularidade, visando auxiliar os donos na recuperação de seus animais de estimação desaparecidos. Por relevância, de acordo com a Figura 1, o sistema aborda a recuperação de cães e gatos.

O sistema permite o cadastramento de anúncios de animais perdidos ou encontrados, a serem divulgados no próprio sistema. Esses anúncios contêm informações relevantes, como momento e local da ocorrência, características do bichinho, dados para contato, fotos e vídeos. Com base nessas informações, o sistema apresenta ao usuário sugestões de anúncios correspondentes com a procura, geradas por um algoritmo de recomendação. Além disso, pode-se pesquisar anúncios cadastrados textualmente, usando suas informações como critérios, e geograficamente, interagindo com um mapa que mostra os locais das ocorrências.

O texto a seguir é dividido em cinco capítulos: o Capítulo 1 apresenta um estudo teórico relevante para a construção do sistema; o Capítulo 2, as principais características de alguns trabalhos relacionados; o Capítulo 3, a motivação, os objetivos, as funcionalidades do sistema em detalhes e algumas de suas telas; o Capítulo 4, o detalhamento do algoritmo de recomendação; o Capítulo 5, a arquitetura e detalhes da implementação do sistema; e o Capítulo 6, testes e análises do algoritmo de recomendação.

⁶ Zoonoses são doenças transmitidas do animal para o homem. Algumas cidades dispõem de órgãos governamentais de controle, os Centros de Controle de Zoonoses (CCZ).

⁷ <<http://www.cachorroperdido.com.br/>>

⁸ <<https://canilvirtual.crowdmap.com/>>

⁹ <<http://procurasecachorro.uol.com.br/>>

1 REFERENCIAL TEÓRICO

Neste capítulo serão abordados alguns conceitos fundamentais para o entendimento deste trabalho. Estes termos estão relacionados à arquitetura do sistema e a funcionalidades que serão apresentadas posteriormente, em detalhes, neste documento.

1.1 HYPERTEXT MARKUP LANGUAGE (HTML)

HTML, ou linguagem de marcação de hipertexto (tradução nossa), é uma linguagem para publicação de conteúdo na *web*. Por sua vez, o termo “hipertexto” define um conjunto de elementos (palavras, mídias, documentos, etc.) interligados, criando uma grande rede de informação (FERREIRA; EIS, 2010, p. 7). Ainda segundo Ferreira e EIS (2010, p. 7), HTML – o idioma – é um dos pilares da *web*, junto com URI¹ – o esquema de nomes – e o HTTP² – o meio de transporte.

A linguagem foi criada originalmente por um físico britânico para disseminar idéias acadêmicas, mas somente a partir de 1997, com a versão 3.2, mantida pelo *World Wide Web Consortium* (W3C), passou a ser considerada um padrão. Desde então, evoluiu pouco, até que, em 2006, com os esforços coordenados do *Web Hypertext Application Technology Working Group* (WHAT *Working Group*, ou simplesmente WHATWG – fundado por Mozilla, Apple e Opera em 2004), trouxe nova vida à *web* através de sua versão atual, o HTML5 (FERREIRA; EIS, 2010, p. 7-8).

O HTML5 incorpora diversas necessidades dos desenvolvedores da *web*, sem a necessidade de *scripts* complexos, como, por exemplo, validação de campos de formulário, armazenamento de dados no lado do cliente independente do servidor (ao contrário dos *cookies*), *drag'n drop* (arrastar-e-soltar), entre outros recursos. Um dos novos recursos, de relevante importância para o nosso trabalho, é o suporte nativo à geolocalização (ou localização geográfica), cujo conceito será apresentado na seção a seguir.

1.2 GEOLOCALIZAÇÃO E GEOCODIFICAÇÃO

Este trabalho utiliza apresentação visual de locais em mapas, o que remete aos conceitos de geolocalização e geocodificação.

A geolocalização, em programação, é a capacidade de obter a posição atual no globo do dispositivo que está sendo utilizado, em coordenadas geográficas (latitude e longitude).

¹ Uniform Resource Identifier, ou identificador uniforme de recursos (tradução nossa).

² Hypertext Transfer Protocol, ou protocolo de transferência de hipertexto (tradução nossa).

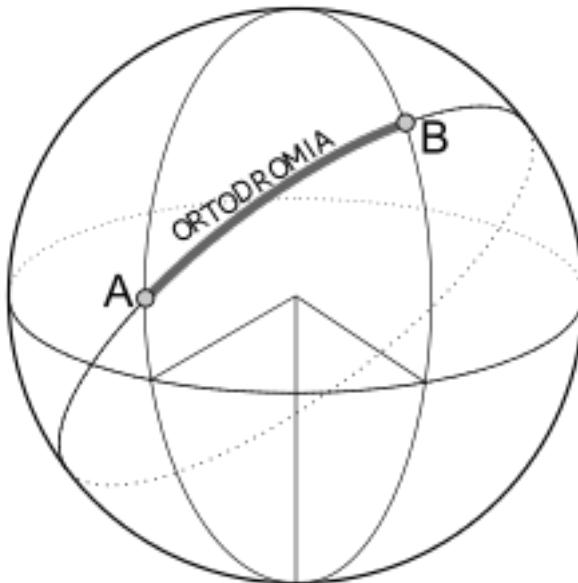
Por sua vez, a geocodificação, conforme Google³ (2013), “é o processo de conversão de endereços [...] em coordenadas geográficas [...]. Ambos os conceitos relacionam-se diretamente à utilização de mapas e marcação de locais em interfaces de usuário, cada vez mais comuns na atualidade.

Neste sistema, os conceitos apresentados são empregados com o uso da API⁴ do Google Maps⁵, em especial a Google Geocoding API⁶, que permite ao usuário trabalhar amigavelmente com locais e não com suas coordenadas.

1.2.1 DISTÂNCIA ENTRE DUAS COORDENADAS GEOGRÁFICAS

O caminho mais curto entre dois pontos na superfície de uma esfera – e aqui, para efeitos práticos, a Terra será considerada uma esfera – é calculado usando a matemática do grande círculo. Um grande círculo é qualquer seção de uma esfera que passe pelo centro. Outras seções são chamadas de pequenos círculos, por definição. Então, o caminho mais curto, ou ortodromia, ou simplesmente a distância entre dois pontos na superfície terrestre é um semiarco (segmento) da circunferência do grande círculo que intercepta os pontos, conforme ilustrado pela Figura 2 (WEISSTEIN, 2002).

Figura 2 – Ortodromia entre os pontos A e B.



Fonte: <https://it.wikipedia.org/wiki/Ortodromia>. Acesso em jun. 2015.

Para achar o arco (S) que representa a distância entre dois pontos (θ_1, λ_1) e (θ_2, λ_2) localizados nas respectivas latitudes (θ) e longitudes (λ), utiliza-se a Equação 1.1

³ <<https://developers.google.com/maps/documentation/geocoding/?hl=pt-br#Geocoding>>

⁴ *Application Programming Interface* ou interface de programação de aplicativos (tradução nossa).

⁵ <<https://developers.google.com/maps/?hl=pt-br>>

⁶ <<https://developers.google.com/maps/documentation/geocoding/>>

(MOREIRA, 2002).

$$\cos(S) = \sin(\theta_1) \times \sin(\theta_2) + \cos(\theta_1) \times \cos(\theta_2) \times \cos(|\lambda_2 - \lambda_1|) \quad (1.1)$$

Uma vez calculado o arco S, em radianos, basta multiplicar pelo raio da Terra para obter a distância linear. O raio a ser utilizado será o da seção equatorial do planeta (Como a Terra é achatada nos polos, o raio real é uma função da latitude.), que é de aproximadamente 6378 quilômetros (WEISSTEIN, 2002).

1.3 SISTEMA DE RECOMENDAÇÃO

Um sistema de recomendação utiliza diversas ferramentas e técnicas computacionais com o objetivo de sugerir, a usuários, itens (serviços ou produtos) que possam ser de seu interesse, baseado no contexto que está inserido, ações realizadas (últimos itens adquiridos ou acessados) ou preferências pessoais (RICCI; ROKACH; SHAPIRA, 2011, p. 1).

Este assunto iniciou como área de pesquisa independente em meados de 1990. No entanto, nos últimos anos, o interesse nessa área vem aumentando drasticamente. Hoje, os sistemas de recomendação desempenham um papel importante em *sites* altamente cotados como Amazon⁷, YouTube⁸, Netflix⁹ e Yahoo¹⁰ (RICCI; ROKACH; SHAPIRA, 2011, p. 3).

Para implementar sua função principal – identificar itens úteis para o usuário – sistemas de recomendação podem utilizar diferentes técnicas. Algumas delas serão explicadas a seguir:

Baseada em conteúdo O sistema aprende a recomendar itens através de outros selecionados pelo usuário em momentos anteriores. A semelhança entre os itens é calculada através de características associativas neles contidas. Por exemplo, se um usuário classificar um filme de comédia positivamente, o sistema pode aprender e recomendar outros filmes do mesmo gênero para aquele usuário (RICCI; ROKACH; SHAPIRA, 2011, p. 11).

Filtragem colaborativa A implementação mais simples desta abordagem visa recomendar a um usuário itens que foram avaliados positivamente por outro usuário com gostos semelhantes. A semelhança entre os gostos dos usuários é calculada a partir do histórico de avaliações dos usuários, inferindo assim que usuários que avaliem os mesmos itens de maneira semelhante tenham alta probabilidade de gostar dos mesmos itens (RICCI; ROKACH; SHAPIRA, 2011, p. 11).

⁷ <<http://www.amazon.com.br>>

⁸ <<https://www.youtube.com>>

⁹ <<https://www.netflix.com>>

¹⁰ <<https://br.yahoo.com>>

Baseada em conhecimento Os itens são recomendados baseado no conhecimento de como as características destes itens são capazes de atender as necessidades preferenciais do usuário, ou utilidade desses itens para o mesmo. Nesta abordagem, a similaridade é calculada em quanto as necessidades do usuário (descrição do problema) coincidem com as recomendações (soluções do problema) (RICCI; ROKACH; SHAPIRA, 2011, p. 12).

O sistema desenvolvido auxilia na procura do dono por seu animal (ou de alguém que encontrou um animal, pelo dono), sugerindo anúncios semelhantes aos que estão sendo acessados, considerando suas informações ou **atributos**. A abordagem mais simples dos sistemas de recomendação **baseados em conteúdo** aplica-se a este problema: quando as preferências do usuário (no caso, seu próprio anúncio ou anúncios vistos) estão descritas em termos do conjunto de atributos dos itens a serem recomendados (anúncios semelhantes), a tarefa de recomendação consiste em comparar os atributos dos itens com as preferências (JANNACH et al., 2010, p. 52-53).

Segundo Jannach et al. (2010), para fazer recomendações, sistemas baseados em conteúdo trabalham calculando a **similaridade** entre itens disponíveis e as preferências do usuário. A similaridade pode ser medida de diversas maneiras. Como exemplo de função de similaridade (s), dado um conjunto de livros ($B = \{B_1, B_2, B_3, \dots\}$) associados cada um a um conjunto de palavras-chave ($k(B_x)$), podemos utilizar o coeficiente de Dice¹¹ para calcular a similaridade entre dois livros B_i e B_j , como na Equação 1.2.

$$s(B_i, B_j) = \frac{2 \times |k(B_i) \cap k(B_j)|}{|k(B_i)| + |k(B_j)|} \quad (1.2)$$

De fato, as métricas utilizadas dependem dos atributos a serem comparados. A similaridade geral entre os itens é dada pela combinação ponderada dos resultados das funções de similaridade (JANNACH et al., 2010).

1.4 WEB SERVICE

Um *web service* é uma aplicação distribuída, cujo seus componentes podem ser executados por dispositivos distintos, normalmente, através de HTTP (KALIN, 2009).

Segundo Chappell e Jewell (2002, p. 1, tradução nossa): “Um *web service* é uma parte da lógica de negócio, localizada em algum lugar na Internet, que é acessível através de protocolos padronizados da Internet, como HTTP ou SMTP¹².”

¹¹ Uma estatística usada para comparar a similaridade entre duas amostras, publicada em 1945 por Lee Raymond Dice.

¹² *Simple Mail Transfer Protocol*, ou protocolo simples de transferência de correio (tradução nossa).

Para este sistema, a utilização de *web services* é de grande importância, visando o reaproveitamento dos serviços oferecidos, tornando-os independentes das plataformas e linguagens utilizadas na interface com o usuário.

1.4.1 REPRESENTATIONAL STATE TRANSFER (REST)

O REST é uma arquitetura escalável para sistemas distribuídos baseada nos protocolos da *web*, em especial o HTTP. As requisições são feitas através de *hyperlinks* e o retorno pode ser qualquer tipo de mídia digital (texto, vídeo, imagem, etc.) (FIELDING, 2000).

REST é utilizado no presente trabalho pela escalabilidade proposta, pela arquitetura e pela simplicidade para a troca de informações, com o uso de protocolos padronizados.

1.4.2 HYPERTEXT TRANSFER PROTOCOL (HTTP)

Segundo a IETF (1999, tradução nossa):

O *Hypertext Transfer Protocol* (HTTP) é um protocolo de nível de aplicação para sistemas de informação distribuídos, colaborativos e de hipermídia. É um protocolo genérico, sem estado, que pode ser usado para muitas tarefas além do hipertexto, como servidores de nome e sistemas de gerenciamento de objetos distribuídos, através da extensão de seus métodos de solicitação, códigos de erro e cabeçalhos.

Devido a escolha de *web service* utilizando a arquitetura REST, o protocolo de transferência que o sistema utiliza para a comunicação entre as interfaces e os serviços é o HTTP, mais especificamente sua versão segura, o *HTTP Over TLS*, mais conhecido como HTTPS¹³. O HTTPS é uma implementação do HTTP sobre uma camada adicional, a *Transport Layer Security* (TLS)¹⁴, sucessora do *Secure Sockets Layer* (SSL)¹⁵, que oferece um canal de comunicação criptografado e verificação de identidade por certificados digitais (IETF, 2000).

1.5 JAVASCRIPT OBJECT NOTATION (JSON)

O JSON, notação de objetos JavaScript (tradução nossa), é um formato leve de troca de informações. É muito utilizado atualmente como formato de dados para troca de informações entre aplicações *web* e móveis com arquitetura cliente-servidor. O formato é simples, textual e facilmente compreendido pelo ser humano, como seu análogo *eXtensible*

¹³ *Hypertext Transfer Protocol Secure*, ou protocolo seguro de transferência de hipertexto (tradução nossa).

¹⁴ Segurança da camada de transporte (tradução nossa).

¹⁵ Camada de soquetes seguros (tradução nossa).

Markup Language (XML). Consiste de duas estruturas básicas: **objetos** – coleções de pares **chave-valor** (entre chaves “{}” e separados por vírgula “,”), também conhecidos como dicionários ou arranjos associativos – e **listas** ordenadas – coleção ordenada de **valores** (entre colchetes “[]”, separados por vírgula), também conhecidos como *arrays* ou vetores. As chaves dos objetos são cadeias de caracteres, ou **strings** (entre aspas duplas “”, enquanto os valores podem ser objetos, listas, **números**, **strings**, **true** (valor lógico “verdadeiro”), **false** (valor lógico “falso”) e **null** (valor nulo) (BRAY, 2014).

Segue um exemplo auto-explicativo de JSON:

```
{
  "pessoas": [
    {"nome": "Francis", "idade": 32, "programador": false},
    {"nome": "Gabriel", "idade": 21, "programador": true}
  ],
  "outras informações": null
}
```

No exemplo acima, os recuos, espaços e quebras de linha servem apenas para facilitar a leitura, pois são ignorados na especificação do formato.

Para este trabalho, optou-se pelo uso do JSON para toda a comunicação entre subsistemas cliente (*web* e móvel) e servidor (*web service*), pela simplicidade e leveza do formato.

1.6 DIFERENÇA ENTRE CORES

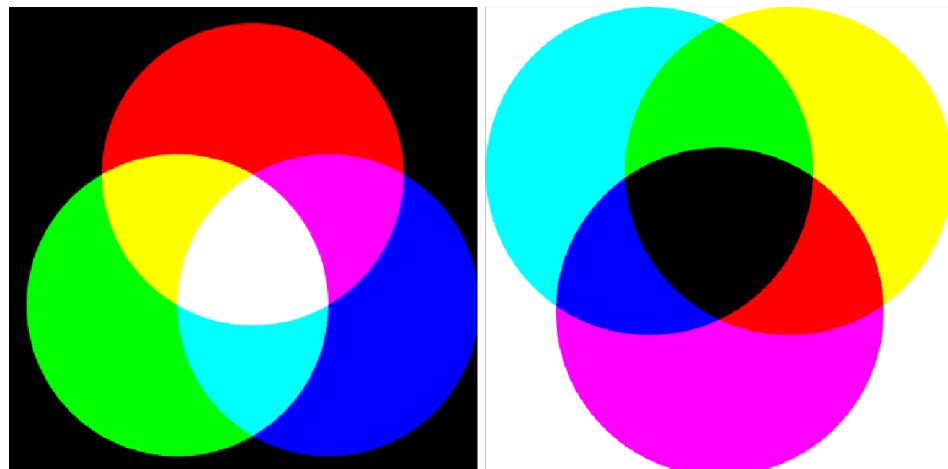
A funcionalidade de sugestão de anúncios semelhantes descrita neste trabalho, seguindo o modelo dos sistemas de recomendação, realiza a comparação subjetiva entre diversas informações contidas nos anúncios de animais perdidos e encontrados, sendo uma delas, a cor.

O conceito de cor não é algo trivial, tampouco a comparação de cores e o estabelecimento da “diferença numérica” entre duas cores quaisquer. Quantificar a diferença entre duas cores é fundamental, de modo a definir claramente qual a contribuição da variável **cor** para a semelhança entre dois anúncios.

A cor é um conceito subjetivo. Consiste em como o cérebro interpreta a informação visual captada pelos receptores visuais (olhos). Tal interpretação é diferente para cada ser vivo dotado de capacidade visual, mesmo os de mesma espécie e de visão completamente saudável. Trata-se, então, de um fenômeno fisiológico, não físico.

Para diferenciar numericamente as cores, antes é necessário defini-las numericamente. Para isso, usa-se um conjunto de coordenadas denominado **espaço ou sistema de cores**. Existem diversos espaços de cores, para diferentes propósitos. Um exemplo bem conhecido é o **sistema RGB**¹⁶, que baseia-se nos tipos de cones¹⁷ da retina humana e define uma cor pela quantidade de “pigmentos” vermelho, verde e azul, também chamadas cores aditivas. A abordagem aditiva é utilizada para produção da sensação de cor através da emissão de luz, como nos monitores de vídeo e televisores, por exemplo. Por outro lado, existe a abordagem subtrativa, na qual a mistura dos pigmentos do espaço de cores produz a sensação de cor através da absorção e reflexão da luz, como no **sistema CMYK**¹⁸, utilizado em impressoras. A Figura 3 ilustra as duas abordagens, através dos dois sistemas.

Figura 3 – Abordagens aditiva (RGB) (esquerda) e subtrativa (CMYK) (direita).



Fonte: Próprios autores (2015).

Apesar da possibilidade de medir a distância entre cores em qualquer sistema, o RGB e o CMYK não refletem como, de fato, o ser humano percebe a diferença entre cores. Uma mesma distância numérica nesses sistemas pode refletir percepções diferentes. Para solucionar a questão, a *Commission Internationale de l'Éclairage* (CIE)¹⁹ definiu, em 1976, o espaço de cores CIELAB (ou $L^*a^*b^*$, ou simplesmente Lab), com foco em uniformidade perceptível, ou seja, que os valores numéricos das diferenças entre quaisquer pares de cores reflitam realmente a diferença percebida pelo ser humano. O sistema Lab é um espaço de coordenadas reais que cobre todo o espectro da luz visível, no qual cada cor é definida pela luminosidade (L^* , no qual 0 indica preto e 100 indica branco), posição entre verde e vermelho (a^* , no qual valores negativos indicam verde e positivos, vermelho) e posição

¹⁶ Red, Green, Blue, ou vermelho, verde e azul - tradução nossa.

¹⁷ Cones são células especiais existentes no olho humano, cada tipo responsável pela interpretação de uma cor básica diferente.

¹⁸ Cyan, Magenta, Yellow, Blac(k), ou ciano, magenta, amarelo e preto - tradução nossa.

¹⁹ Comissão Internacional de Iluminação, tradução nossa.

entre azul e amarelo (b^* , no qual valores negativos indicam azul e positivos, amarelo) (ROBERTSON, 1977).

Por fim, Robertson (1977) definiu, também em 1976, o Delta-E (ΔE), grandeza que representa a diferença numérica entre cores, de acordo com a percepção humana. Se considerarmos cores como pontos no espaço – no caso, o espaço de cores Lab – o cálculo do ΔE é dado pela distância euclidiana entre dois pontos (L_1, a_1, b_1) e (L_2, a_2, b_2) , como ilustrado pela Equação 1.3.

$$\Delta E = \sqrt{(L_2 - L_1)^2 + (a_2 - a_1)^2 + (b_2 - b_1)^2} \quad (1.3)$$

O ΔE é utilizado como referência para o cálculo de similaridade entre cores no algoritmo de recomendação, a ser detalhado no Capítulo 4.

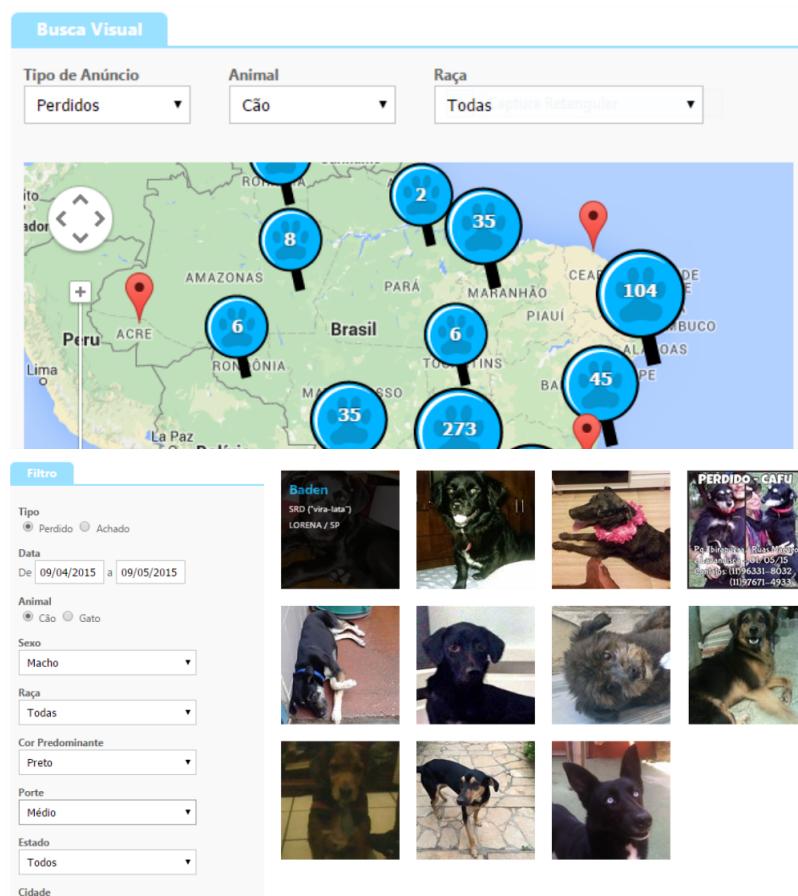
No capítulo a seguir, serão apresentados alguns detalhes de alguns trabalhos relacionados ao tema abordado neste documento.

2 TRABALHOS RELACIONADOS

Neste capítulo, serão apresentadas algumas ferramentas para divulgação, de animais perdidos e encontrados, através da Internet, conforme mencionado na Introdução. São elas o Cachorro Perdido, o Canil Virtual e o Procura-se Cachorro.

O Cachorro Perdido é uma aplicação *web* que permite o cadastro de anúncios de animais perdidos, encontrados e para doação, com dados para contato, data e local do ocorrido, uma foto e características do animal. Apresenta, na página inicial, um mapa com marcadores indicando, na região exibida, o local das ocorrências cadastradas e permite filtrá-las por tipo de anúncio (perdidos ou achados), espécie (cão ou gato) e raça. Através da opção “Achados & Perdidos”, exibe uma lista de fotos dos animais anunciados e permite filtrá-las por período, espécie, gênero, raça, cor predominante, porte, estado e cidade. Ao clicar nos marcadores ou nas fotos, são exibidos os detalhes do anúncio. As telas de busca são ilustradas na Figura 4.

Figura 4 – Busca de anúncios no Cachorro Perdido.



Fonte: Cachorro Perdido (2015).

O Canil Virtual é outra aplicação *web* que permite o cadastro de “relatos”, que contém um título, uma descrição, a data, a hora e o local. Devem enquadrar-se em uma ou mais categorias: cães e gatos desaparecidos ou encontrados, abandonados, para adoção, ONGs e protetores, clínicas populares, notícias, pedidos de ajuda e finais felizes. Pode-se anexar uma ou mais fotos e *links* de vídeos e notícias. Uma desvantagem dos relatos com relação aos anúncios do Cachorro Perdido é não poderem ser classificados pelas características do animal, exceto o gênero, que é uma subcategoria. Além do formulário próprio, o Canil Virtual permite cadastrar relatos por *e-mail*, SMS e Twitter. Apesar de apresentar o mapa com marcadores e listas de relatos (Figura 5), como a aplicação anterior, os filtros são diferentes: período, categoria e subcategoria, local, meio de envio, tipo de mídia anexada (fotos, vídeos ou notícias) e estado da “verificação” (verificado ou não – a verificação não é explicada no *site*). Uma funcionalidade interessante é a possibilidade de receber alertas por *e-mail* para cada novo relato, de acordo com seu local e categoria: o usuário informa um endereço de *e-mail*, define uma região circular no mapa (com raio de 1 a 100 km) e escolhe as categorias desejadas (Figura 6).

Figura 5 – Lista de relatos no Canil Virtual.

Filtrar Relatos Por

Categoria	Limpar
Todas as categorias	814
Cão ou gato abandonado	53
Macho	30
Fêmea	23
Cão ou gato para adoção	521
Macho	246
Fêmea	275
Cão ou gato desaparecido	146
Macho	71
Fêmea	75

Local Limpar ▾

Tipo Limpar ▾

Mídia Limpar ▾

Verificação Limpar ▾

Fonte: Canil Virtual (2015).

O Procura-se Cachorro é uma aplicação híbrida – *web* e móvel (Apple iOS¹) – que permite o cadastro de “ocorrências” de (apenas) cães perdidos, encontrados ou para adoção. As ocorrências são descritas por seu local (endereço ou nome do local), “situação” (perdido, achado ou para adoção), características e fotos do cão, data e detalhes. Mais do que somente uma aplicação, é um portal de informações com notícias, artigos, etc. Assim como as aplicações descritas anteriormente, oferece marcações em um mapa e busca por palavra-chave, situação, raça, cor e gênero (Figura 8). Uma funcionalidade interessante é

¹ <<https://www.apple.com/br/ios/what-is/>>

Figura 6 – Cadastro para recebimento de alertas no Canil Virtual.

Receber alertas

Primeiro passo: Selecione sua cidade ou localidade

Ou defina um local no mapa abaixo, e você receberá um alerta quando forem feitos relatos em um raio de 20 quilômetros.

Segundo passo: Enviar alertas para meu

Email:
forneça um email

Terceiro passo (opcional): Escolha categorias

- Cão ou gato abandonado
- Cão ou gato para adoção
- Cão ou gato desaparecido
- Cão ou gato encontrado
- ONGs e Protetores
- Clínica popular
- Pedido de ajuda
- Notícias
- Finais Felizes!

Salvar meus alertas

Confirmar solicitação anterior de alerta

Fonte: Canil Virtual (2015).

a seção “este animal pode ser o que procura” no rodapé da tela que mostra os detalhes de uma ocorrência (Figura 7). Essa seção mostra sugestões de ocorrências semelhantes à detalhada no momento, porém de “situação” oposta. Por exemplo, se acessados os detalhes de uma ocorrência de cão perdido, as sugestões serão de encontrados e vice-versa. A aplicação móvel do Procura-se Cachorro (Figura 9) conta com menos funcionalidades que a *web*: não há busca textual, nem mesmo para centralizar o mapa em outro local (inicia mostrando o local atual do dispositivo, se disponível), e não há como gerenciar (editar, remover, etc.) as próprias ocorrências cadastradas. No entanto, quando vários marcadores são exibidos no mapa, ao tocar em um, acessando os detalhes daquela ocorrência, botões de navegação (anterior e próxima) dão acesso às demais, sem precisar voltar ao mapa.

A avaliação das aplicações citadas anteriormente serviu como base para a escolha das funcionalidades deste trabalho, detalhadas no Capítulo 3.

Figura 7 – Ocorrência em detalhes no Procura-se Cachorro.

LALA/LALINHA - MONTE ALEGRE, TABOÃO DA SERRA/SP



RAÇA: Basset/Dachshund
NÚMERO DA MEDALHA DO PROCURA-SE CACHORRO:
 0
GÊNERO: Fêmea
COR: Preto
PERDIDO EM: 18/04/2015
ENDEREÇO: argentina - monte alegre, Taboão da Serra/SP
CONTATO: [REDACTED]
E-MAIL: [REDACTED]
MAIS INFORMAÇÕES: contato c/ [REDACTED]

[Gerar cartaz](#)

1

LOCALIZAÇÃO





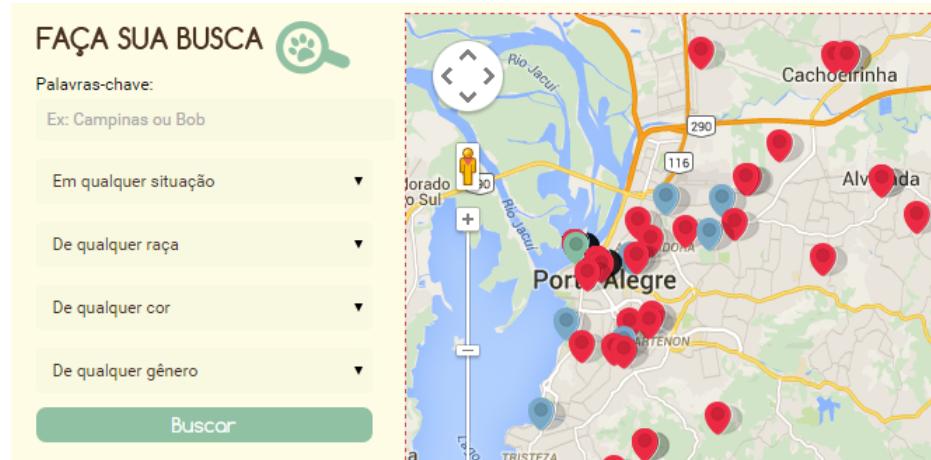
Nome:
NÃO SABEMOS
Raça:
BASSET/DACH...
Encontrado em:
SÃO PAULO/SP



Nome:
NAO SEI
Raça:
BASSET/DACH...
Encontrado em:
SÃO PAULO/SP

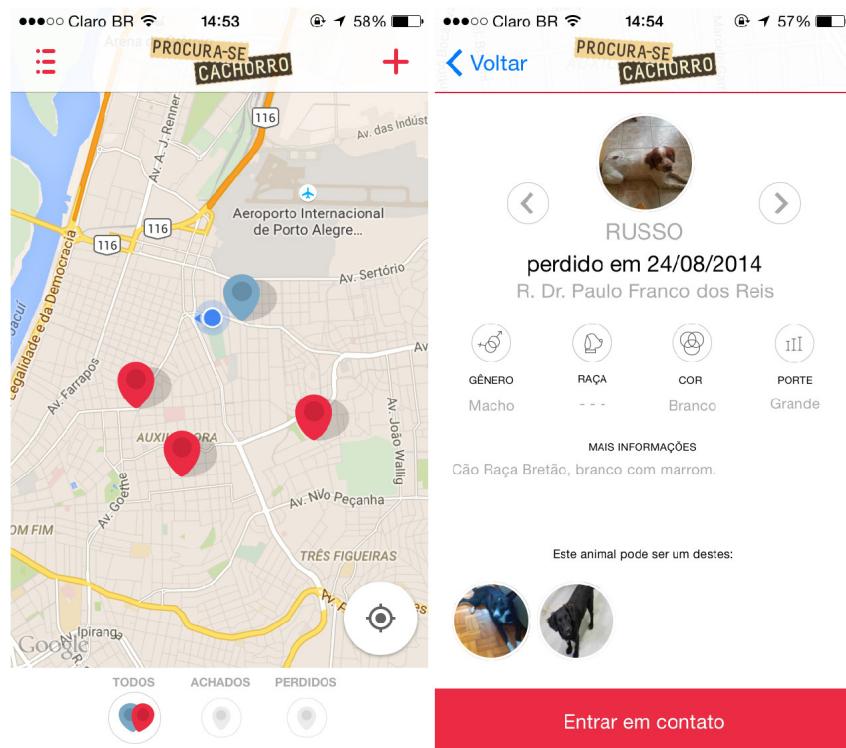
Fonte: Procura-se Cachorro (2015).

Figura 8 – Busca no Procura-se Cachorro.



Fonte: Procura-se Cachorro (2015).

Figura 9 – Telas do aplicativo móvel Procura-se Cachorro.



Fonte: Procura-se Cachorro (2015).

3 VISÃO GERAL DO SISTEMA

Neste capítulo, será apresentado o nosso sistema: a motivação, os objetivos, detalhes do funcionamento do sistema e imagens de suas telas.

3.1 MOTIVAÇÃO E OBJETIVO GERAL

Este trabalho é motivado pela causa animal. Diversos são os problemas relacionados a essa causa: maus tratos, abandono e exploração comercial são alguns exemplos. Apesar do cenário cruel, muitos animais, em especial cães e gatos, têm a sorte de terem um lar, uma família, serem bem cuidados, com carinho e atenção. A recompensa para quem opta por ter um bichinho em casa é muito grande: os animais são leais, são ótimas companhias e sabem retribuir o amor de um jeito muito especial. Em contrapartida, vivem relativamente pouco – em torno de 12 anos – e, com uma certa facilidade, se perdem. A lacuna deixada por um membro da família perdido é irreparável. Nesse sentido, acreditando-se no papel social da Tecnologia da Informação (TI), pretendeu-se contribuir com conhecimento e experiência para a solução do problema.

O objetivo deste trabalho foi consolidar e ampliar os conhecimentos adquiridos durante o Bacharelado em Sistemas de Informação, pesquisando e aprendendo como as tecnologias e os recursos computacionais disponíveis atualmente podem contribuir para essa causa tão delicada. Pretendeu-se construir um sistema que fosse efetivo na solução de casos de perda de animais, auxiliando na divulgação e na busca por cães e gatos perdidos ou encontrados e focou-se no suporte ao navegador *web* Google Chrome, versões *desktop* e para o sistema operacional Android.

3.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos do trabalho foram os seguintes:

- pesquisar métodos e algoritmos de recomendação;
- elaborar regras do sistema de sugestão;
- estudar plataformas, linguagens e tecnologias *web* e móvel;
- modelar e desenvolver o sistema **Bicho Perdido**, com as seguintes funcionalidades:
 - registro e autenticação de usuários;

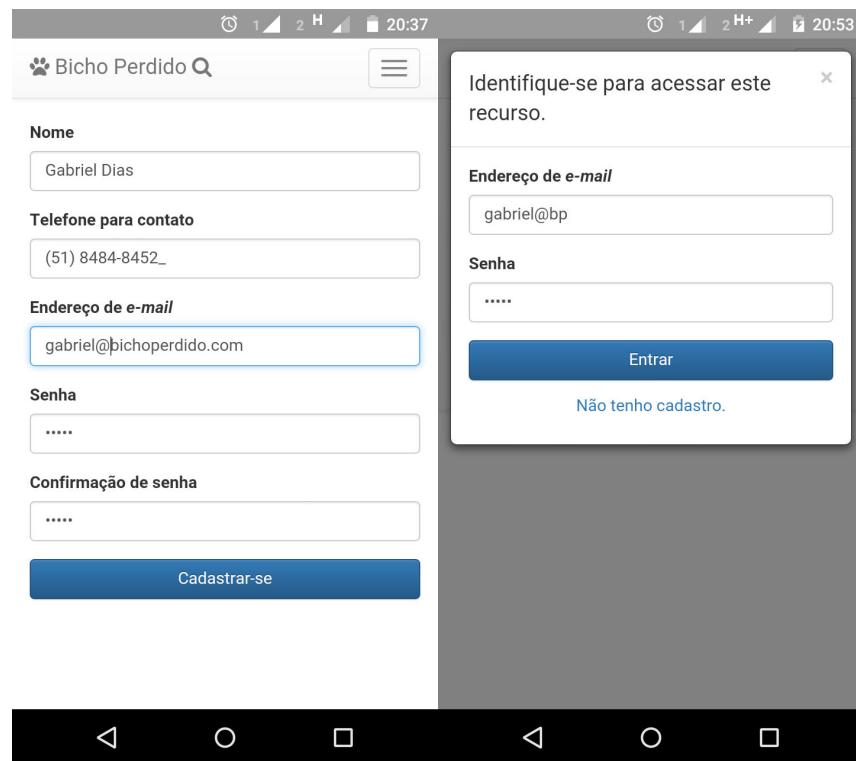
- cadastro de anúncios de animais perdidos e encontrados;
- busca textual e geográfica por anúncios;
- sugestão de anúncios semelhantes;
- notificação de usuários sobre anúncios semelhantes;
- integração com redes sociais.

A seguir, serão detalhadas as funcionalidades listadas anteriormente.

3.3 FUNCIONAMENTO DO SISTEMA EM DETALHES

O sistema permite o registro de usuários, com informações para contato (nome, telefone e *e-mail*) e uma senha de acesso, bem como a autenticação (identificação por endereço de *e-mail* e senha) de usuários, ilustrado pela Figura 10, que dá acesso a funcionalidades que serão explicadas a seguir.

Figura 10 – Telas de criação e autenticação de usuários.



Fonte: Próprios autores (2015).

Usuários autenticados podem cadastrar anúncios de animais perdidos ou encontrados, conforme ilustrado pela Figura 11, incluindo informações sobre a ocorrência – natureza (perdido ou encontrado), data, hora, local (coordenadas geográficas – latitude e longitude) e detalhes julgados relevantes (“perdeu-se na rua”, “fugiu de casa”, etc.) –

e as características do animal – espécie (cão ou gato), nome, gênero, cores, raça, porte (pequeno, médio e grande, para cães), pelagem (curta e longa), peculiaridades (como manchas e cicatrizes) e outras (como adereços e comportamentos), bem como anexarem fotos e endereços de vídeos no YouTube¹. Um novo anúncio é considerado “em aberto”, até que sua ocorrência seja solucionada, quando então passa a ser considerado “resolvido”.

Sendo de extrema importância, o local da ocorrência pode ser informado de diversas formas: endereço ou nome do local, ponto no mapa, localização atual (para dispositivos móveis ou equipados com GPS²) ou coordenadas de geolocalização (latitude e longitude) disponíveis nos metadados do arquivo de imagem principal do anúncio (foto de capa). Para a apresentação visual dos locais, utilizou-se a API do Google Maps, conforme ilustrado pela Figura 11.

Além disso, o usuário deve indicar as cores do animal, de uma a três, por ordem de predominância, usando um seletor de cores, conforme ilustrado pela Figura 11. No projeto do sistema, pretendia-se utilizar um conjunto pré-definido de cores para facilitar o preenchimento, conforme ilustrado pela Figura 12. Porém, optou-se por utilizar todas as cores possíveis para demonstrar o quanto a comparação utilizando o Delta-E (ΔE) é genérica e eficiente.

As peculiaridades do animal são informadas a partir de combinações de duas variáveis: “O quê?” e “Onde?”. Os valores possíveis dessas variáveis são disponibilizados em listas pré-definidas. Dentre os valores possíveis para “O quê?”, tem-se: cicatriz, deficiência, falha e mancha. Os valores possíveis para “Onde?” são: cabeça, orelha (esquerda ou direita), tronco (dorso ou torso), pata (frontal ou traseira, esquerda ou direita) e cauda, conforme ilustrado pela Figura 11. É possível informar mais de uma peculiaridade. As peculiaridades são utilizadas para comparar anúncios semelhantes.

O sistema possui nomenclatura especial para cada tipo de usuário, conforme segue:

Perdedor É o usuário autenticado que cadastrou pelo menos um anúncio de animal perdido, ainda em aberto.

Achador É o usuário autenticado que cadastrou pelo menos um anúncio de animal encontrado, ainda em aberto.

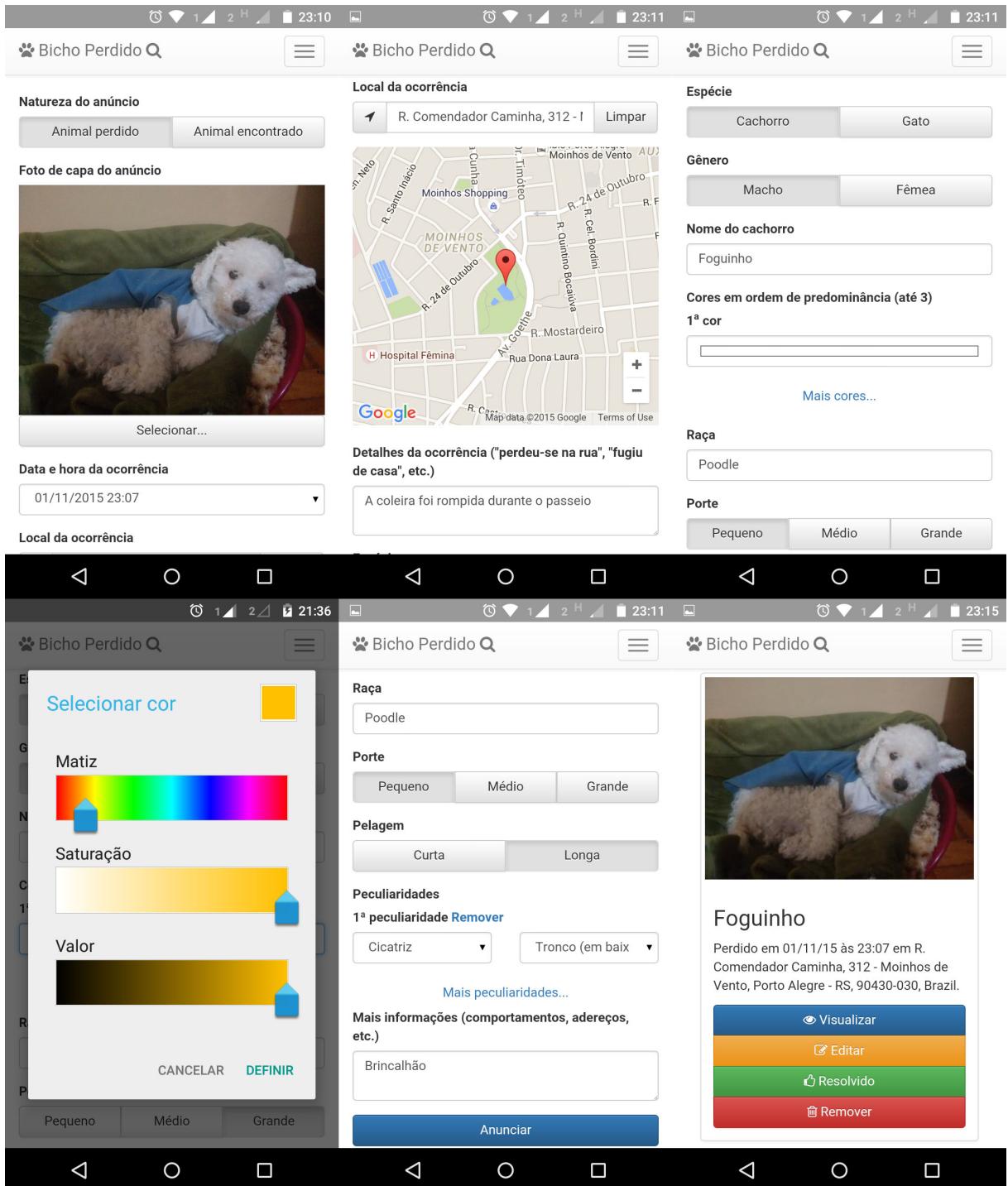
Protetor É um tipo especial de usuário autenticado, destinado a ONGs, *pet shops* e voluntários que desejam acompanhar novos registros de ocorrências em uma determinada região.

Visitante É o usuário não autenticado.

¹ Serviço de compartilhamento de vídeos, disponível em <<https://www.youtube.com/>>.

² Global Positioning System ou sistema de posicionamento global (tradução nossa).

Figura 11 – Telas de criação de um anúncio.

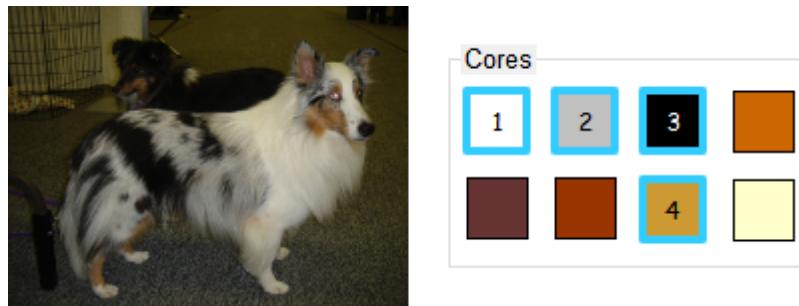


Fonte: Próprios autores (2015).

Qualquer usuário autenticado pode colaborar como protetor, bastando definir a região que deseja monitorar, chamada de “área de cobertura”. A região é circular e definida por um ponto no mapa (centro) e o raio em quilômetros, conforme ilustrado pela Figura 13.

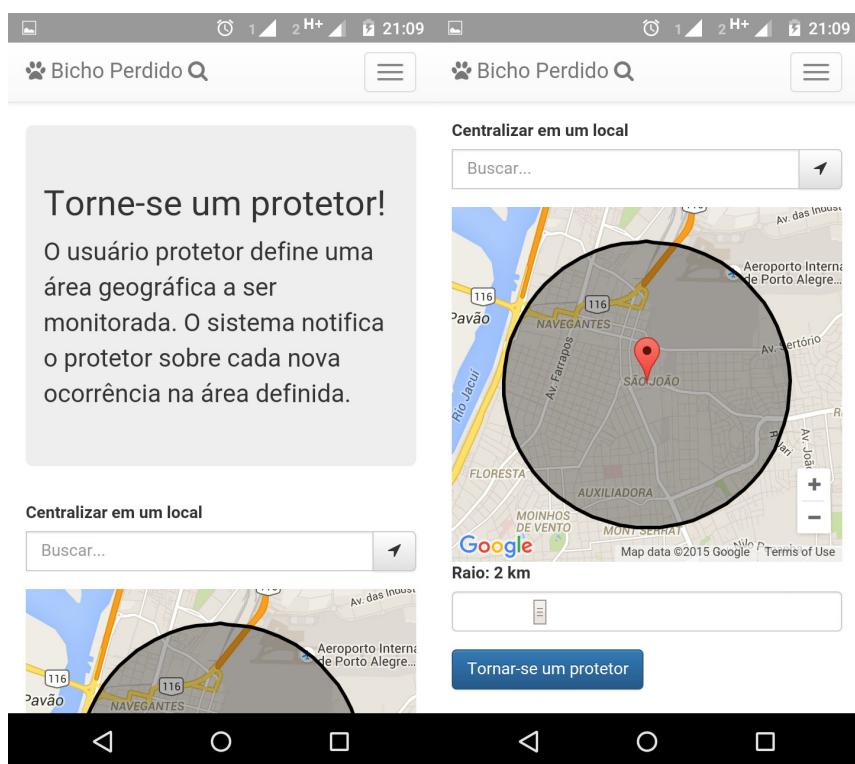
A busca por anúncios pode ser realizada por qualquer usuário do sistema, inclusive visitantes, de duas formas diferentes. É possível buscar por palavras-chave, obtendo como

Figura 12 – Esboço de seleção de cores pré-definidas.



Fonte: Próprios autores (2015).

Figura 13 – Telas de definições de usuário protetor.

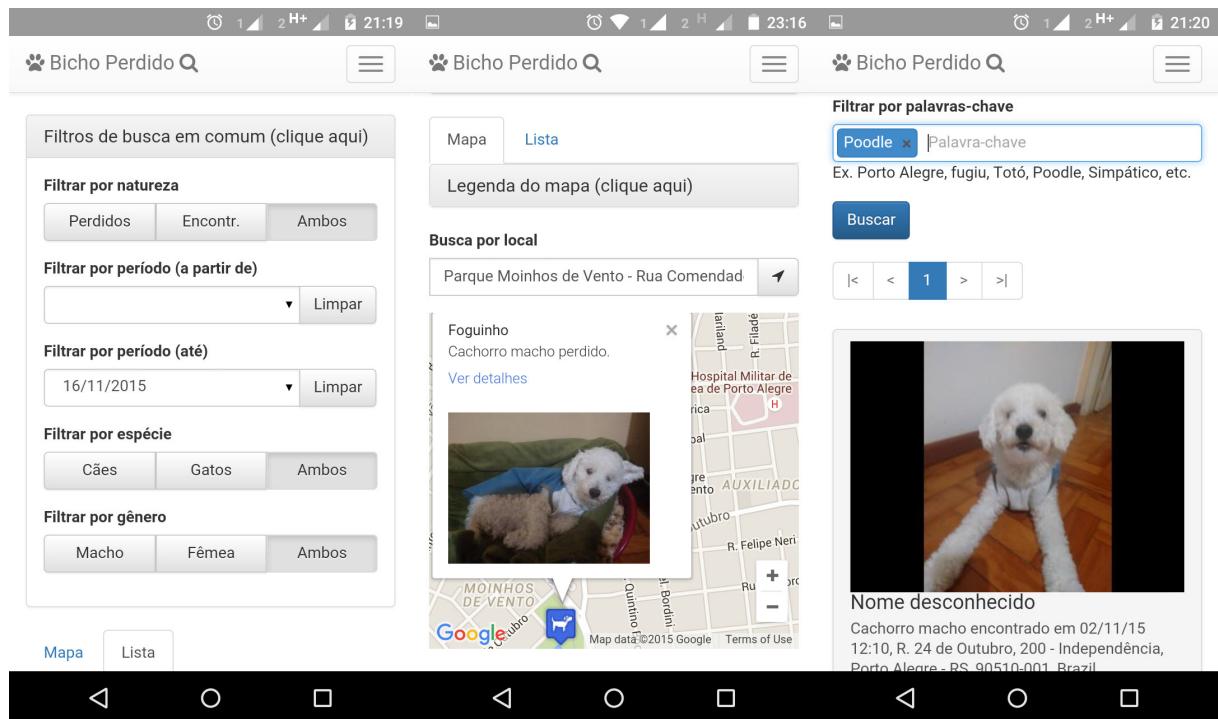


Fonte: Próprios autores (2015).

resultado uma lista com os anúncios correspondentes. Define-se com isso a “busca textual”. Além disso, há a “busca geográfica”, que consiste em interagir com um mapa com marcadores indicando os locais cadastrados nos anúncios. Esses marcadores são *links* que exibem os detalhes do anúncio relacionado. É permitido ao usuário aumentar ou diminuir o nível de detalhe (*zoom*) desse mapa. Todos os anúncios cujo local pertença à região exibida na área do mapa são mostrados. A busca geográfica utiliza a tecnologia Google Maps. Existem ainda filtros em comum, que se aplicam a ambos os tipos de busca. É possível filtrar os resultados por natureza do anúncio, período, espécie do animal e gênero, conforme ilustrado pela Figura 14. Por fim, a partir dos detalhes de qualquer anúncio, o sistema oferece a opção de gerar automaticamente e imprimir cartazes para divulgação na rua,

conforme ilustrado pela Figura 15.

Figura 14 – Telas de busca por anúncios.



Fonte: Próprios autores (2015).

Figura 15 – Telas de geração de cartazes.

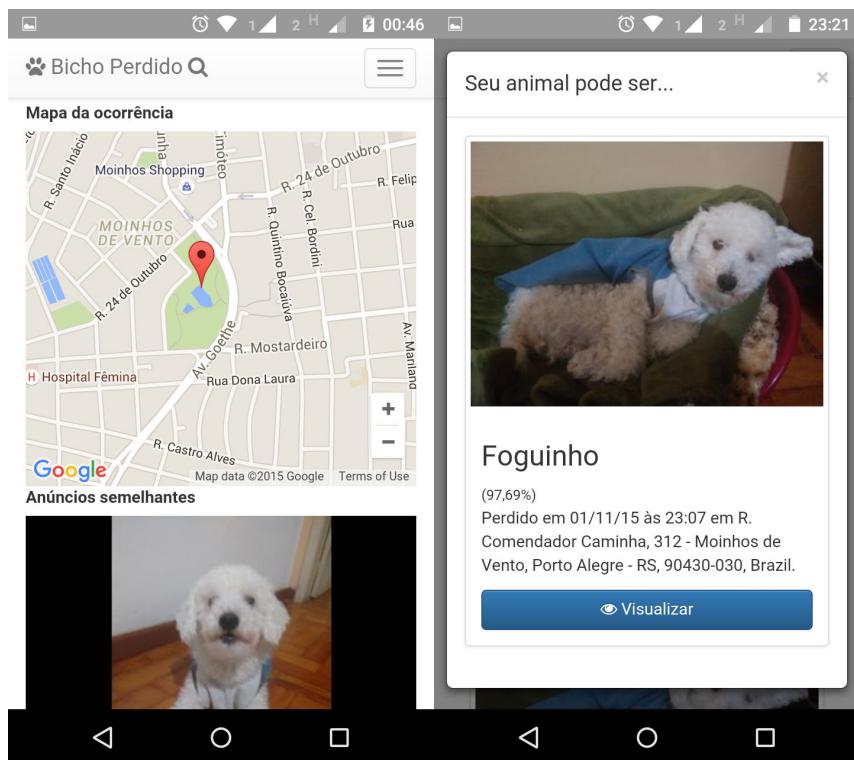


Fonte: Próprios autores (2015).

O sistema faz sugestões – geradas pelo algoritmo de recomendação – de anúncios correspondentes ao que o usuário procura, em dois momentos distintos: ao cadastrar

um novo anúncio e ao exibir os detalhes de um anúncio cadastrado, conforme ilustrado pela Figura 16. No primeiro momento, usuários “perdedores” veem anúncios de animais encontrados – e “achadores”, de perdidos – semelhantes ao seu. No segundo momento, são sugeridos anúncios semelhantes de mesma categoria. O algoritmo de recomendação é baseado em um sistema de pesos, para que informações divergentes não sejam um critério de exclusão, estabelecendo tolerância a erros de preenchimento. Por exemplo, a Figura 17 mostra um cachorro que, ao ser cadastrado como perdido, poderia ser classificado pelo dono como de cor “creme”. No entanto, mesmo se encontrado posteriormente e cadastrado como de cor “branca”, deve fazer parte das sugestões, pois trata-se do mesmo cão. As regras exatas estão definidas no Capítulo 4.

Figura 16 – Telas de sugestões de anúncios semelhantes.



Fonte: Próprios autores (2015).

A informação do nome do animal, no caso de cadastro de anúncio de animal encontrado, somente deve ser preenchida em caso de certeza, ou seja, se o animal estava devidamente identificado quando foi achado, por medalha ou coleira. Informar o nome incorretamente impacta o sistema de recomendação.

Sempre que um novo anúncio é cadastrado, o sistema emite notificações sobre a existência desse novo anúncio – pelo navegador via API de notificações do HTML5 – aos usuários que tiveram seu anúncio sugerido, nesse momento, pelo algoritmo de recomendação,

³ <<http://canilasgardbull.blogspot.com.br/2014/10/locus-e-vermelho-dourado-e-creme.html>>. Acesso em maio de 2015.

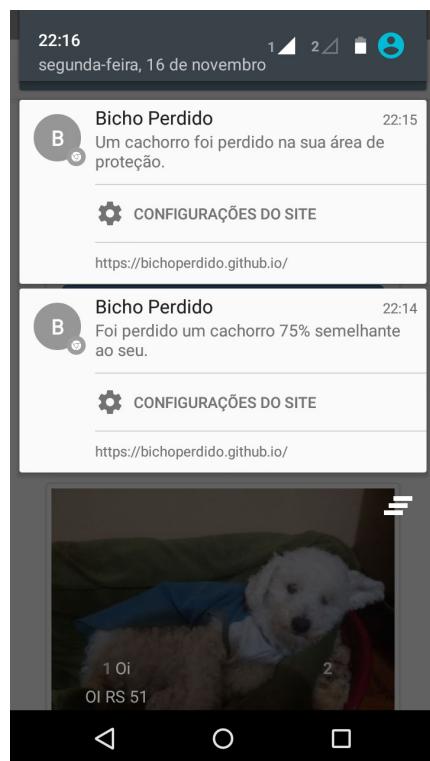
Figura 17 – Exemplo de cão com cor duvidosa.



Fonte: Canil AsgardBull³ (2014).

explicado anteriormente; e aos usuários protetores, caso o local do novo anúncio pertença à sua área de cobertura, conforme ilustrado pela Figura 18.

Figura 18 – Tala com notificações recebidas.



Fonte: Próprios autores (2015).

Por fim, de acordo com o projeto, o sistema deveria ser integrado com algumas redes sociais para ampliar a divulgação. Seria oferecido ao usuário a possibilidade de compartilhar o anúncio, com sua permissão, através de suas contas no Facebook, no Twitter e no Instagram. Além disso, seria criada uma página no Facebook e perfis no Twitter e no Instagram próprios do sistema. Nesses canais, o compartilhamento seria automático. Para facilitar o registro do usuário, o sistema permitiria autenticação – e registro automático – com as mesmas redes sociais e também com o Google⁴. Porém, não houve tempo suficiente para implementar essa funcionalidade.

A Tabela 4 ilustra a comparação entre o Bicho Perdido e os trabalhos relacionados analisados no capítulo anterior.

Tabela 4 – Comparativo de ferramentas para divulgação de animais perdidos.

	Cachorro Perdido	Camil Virtual	Procura-se Cachorro	Bicho Perdido
Interface web	Sim	Sim	Sim	Sim
Aplicativo móvel	Não	Não	Sim	Sim
Cães e gatos	Sim	Sim	Não	Sim
Buscar por critérios	Sim	Sim	Sim	Sim
Buscar no mapa	Sim	Sim	Sim	Sim
Geração de cartazes	Não	Não	Sim	Sim
Monitorar uma área	Não	Sim	Não	Sim
Sugestão de anúncios semelhantes	Não	Não	Sim	Sim
Notificações	Não	Sim	Não	Sim
Integração com redes sociais	Sim	Sim	Sim	Não

Fonte: Próprios autores (2015).

⁴ <<http://www.google.com.br>>

4 ALGORITMO DE RECOMENDAÇÃO

Neste capítulo é detalhado o algoritmo que recomenda anúncios semelhantes ao usuário do sistema, com base na comparação ponderada das informações dos anúncios – as quais neste escopo são chamadas de **atributos** – seguindo o modelo dos sistemas de recomendação baseados em conteúdo.

O algoritmo faz a comparação de anúncios, dois a dois, atributo por atributo. Cada atributo é comparado por uma fórmula específica, resultando em seu grau de semelhança, de 0 (zero) a 100% (cem por cento). A média ponderada das semelhanças entre os atributos define a semelhança entre dois anúncios, também de 0 a 100%. A seguir, serão relacionados e classificados os atributos utilizados na comparação.

4.1 SEMELHANÇA POR ATRIBUTO

Por definição, os atributos são classificados em três tipos:

Normal A semelhança entre dois valores de um atributo deste tipo **sempre** participa da média ponderada final que define a semelhança resultante entre dois anúncios.

Exclusivo Se a comparação entre valores de qualquer atributo Exclusivo resultar em semelhança zero, a semelhança resultante entre os anúncios também é zero, independente dos demais atributos. Se a semelhança for maior, o atributo comporta-se como o tipo Normal.

Bônus O grau de semelhança de atributos Bônus somente influencia positivamente na média final. Se sua influência for negativa, o atributo é desconsiderado.

Cada atributo possui um peso de 1 a 3 indicando sua relevância para a semelhança entre dois anúncios. A Tabela 5 relaciona os pesos numéricos com o grau de relevância (linguístico).

Tabela 5 – Relação do peso do atributo com grau de relevância.

Peso	Grau de relevância
1	Pouco relevante
2	Relevante
3	Muito relevante

Fonte: Próprios autores (2015).

Um dos atributos utilizados para a comparação é o conjunto de **cores** do animal descrito no anúncio. Esse atributo é classificado como **exclusivo** de **peso 3**. Cada cor do conjunto de um anúncio é comparada com todas as cores do conjunto do outro anúncio, resultando em um grau de semelhança de 0 a 100%, dispostos em uma matriz como no exemplo da Figura 19. Na Figura 19, a semelhança entre a cor 1 do anúncio A e a cor 1 do anúncio B é de 10%, entre a cor 2 do anúncio A e a cor 1 do anúncio B é de 30% e assim por diante.

Figura 19 – Exemplo de matriz de semelhança entre cores de um anúncio com 3 cores e outro com 2.

		Anúncio A		
		Cor 1	Cor 2	Cor 3
Anúncio B	Cor 1	10%	30%	80%
	Cor 2	50%	70%	10%

Fonte: Próprios autores (2015).

Para o cálculo do grau de semelhança entre duas cores, é utilizado o Delta-E (ΔE), grandeza descrita no Capítulo 1. O ΔE é um número real que representa a diferença percebida entre duas cores. Quanto maior o ΔE , maior a diferença. Após testes com diversos pares de cores, convencionou-se considerar ΔEs maiores ou iguais a 50 como 0% de semelhança. Então, o cálculo do grau de semelhança (S) está representado na Equação 4.1.

$$S = \begin{cases} \frac{50 - \Delta E}{50} & \text{se } \Delta E < 50 \\ 0 & \text{se } \Delta E \geq 50 \end{cases} \quad (4.1)$$

Para os casos de comparação de anúncios com mais de uma cor, após calcular a matriz (Figura 19), a semelhança resultante do atributo cor é definida da seguinte forma:

1. Seleciona-se o maior grau da matriz. No exemplo da Figura 20, o maior grau é 80%;

Figura 20 – Selecionando o maior grau da matriz.

		Anúncio A		
		Cor 1	Cor 2	Cor 3
Anúncio B	Cor 1	10%	30%	80%
	Cor 2	50%	70%	10%

Fonte: Próprios autores (2015).

2. Remove-se da matriz a linha e a coluna correspondente ao grau selecionado, gerando uma nova matriz menor. No exemplo da Figura 21, removeu-se a linha correspondente

à cor 1 de B e a coluna da cor 3 de A, sobrando uma matriz com os valores 50% e 70%;

Figura 21 – Removendo a linha e a coluna correspondente ao maior grau.

		Anúncio A		
		Cor 1	Cor 2	Cor 3
Anúncio B	Cor 1	10%	30%	80%
	Cor 2	50%	70%	10%

Fonte: Próprios autores (2015).

3. Repete-se o procedimento com a nova matriz enquanto houver pelo menos um elemento. No exemplo da Figura 22, selecionou-se 70% e descartou-se o resto, terminando o procedimento;

Figura 22 – Repetindo o procedimento para a nova matriz.

		Anúncio A	
		Cor 1	Cor 2
Anúncio B	Cor 2	50%	70%

Fonte: Próprios autores (2015).

4. O grau resultante é a média aritmética simples dos graus selecionados. No exemplo da Figura 23, é 75% ou 0,75.

Figura 23 – Calculando o grau resultante do atributo cor.

$$\text{Grau de semelhança resultante} = \frac{80\% + 70\%}{2} = 75\% = 0,75$$

Fonte: Próprios autores (2015).

No caso da matriz conter o grau mais alto repetido em uma mesma linha ou coluna, seleciona-se o mais próximo da diagonal da matriz, ou seja, respeitando a predominância das cores no anúncio, conforme ilustrado pela Figura 24. Caso o grau mais alto repita-se em linhas e colunas diferentes, o resultado final é o mesmo, independente da ordem de seleção entre eles.

Por fim, consideremos as comparações dos conjuntos de cores a seguir:

- preto, marrom e branco com as mesmas cores;
- e preto, marrom e branco com apenas a cor preta.

Figura 24 – Selecionando o maior grau repetido mais próximo da diagonal da matriz.

		Anúncio A		
		Cor 1	Cor 2	Cor 3
Anúncio B	Cor 1	10%	30%	80%
	Cor 2	50%	70%	(80%)

Fonte: Próprios autores (2015).

Ambas comparações resultam em um grau de semelhança de 100%. Para corrigir essa discrepância, resolveu-se penalizar o grau resultante do atributo em 10% para cada cor a mais (ou a menos) em um dos conjuntos. Voltando ao exemplo da Figura 23, o grau de semelhança calculado (75%) é penalizado em 10% porque o Anúncio A tem uma cor a mais que o Anúncio B (ou o Anúncio B tem uma cor a menos). A correção resulta no grau 67,5% ou 0,675 e é ilustrada pela Figura 25.

Figura 25 – Corrigindo o grau resultante do atributo cor.

$$\text{Grau de semelhança corrigido} = 75\% \times (100\% - 10\%) = 67,5\% = 0,675$$

Fonte: Próprios autores (2015).

Outro atributo utilizado é o **momento** (data e hora) da ocorrência (animal perdido ou encontrado) registrada no anúncio. Diferente dos outros atributos, ele é utilizado apenas para comparação entre anúncios de naturezas distintas (perdido com encontrado e vice-versa) e é classificado como atributo **exclusivo** de **peso 2**. Quando o anúncio de animal perdido é posterior (data e hora posteriores) a outro anúncio de animal encontrado, o grau de semelhança entre eles é 0%. Caso contrário, é 100%. O momento é um atributo exclusivo porque não faz sentido sugerir animais encontrados antes de serem perdidos e é comparado dessa forma porque não deseja-se considerar o tempo que o animal está perdido como penalidade.

O **local** das ocorrências anunciadas também é atributo e são consideradas, para comparação, as estatísticas apresentadas nas Tabelas 2 e 3. Esse atributo é do tipo **normal** de **peso 2**. Após o cálculo da distância linear, utilizando a matemática do grande círculo, explicada no Capítulo 1, converte-se a distância em grau de semelhança usando as Tabelas 6 e 7, para cães e gatos respectivamente.

Ambos os atributos **gênero** e **nome** são classificados como **exclusivos de peso 3**. Seus graus de semelhança são binários: 0% ou 100%. Valores iguais significam 100% de semelhança e diferentes, 0%, com exceção para o valor “não sei”: qualquer valor comparado com “não sei” resulta em 100%. Para que a comparação faça sentido, presume-se a veracidade das informações prestadas pelos usuários.

Tabela 6 – Grau de semelhança para o atributo local (distância) entre cães.

Distância (km) maior que	Grau de semelhança
50	1%
25	3%
15	6%
10	11%
5	22%
2	51%
0	100%

Fonte: Próprios autores (2015).

Tabela 7 – Grau de semelhança para o atributo local (distância) entre gatos.

Distância (m) maior que	Grau de semelhança
1500	6%
500	14%
300	25%
150	39%
0	100%

Fonte: Próprios autores (2015).

O conjunto de **peculiaridades** do animal é um atributo **bônus** de peso variável. Excepcionalmente neste atributo, o peso tem significado exclusivamente numérico. Nenhuma peculiaridade comum resulta em grau de semelhança 0%. Pelo menos uma comum resulta em 100%. Para cada peculiaridade comum entre os conjuntos em comparação, o peso aumenta em uma unidade. Por exemplo, se existem duas peculiaridades em comum, o grau será de 100% e o peso será 2. O atributo foi definido como bônus por ser possível que cicatrizes ou manchas, por exemplo, não sejam notadas pelo dono do anúncio ou tenham sido adquiridas pelo animal durante o tempo que esteve perdido.

Os atributos **porte** e **pelagem** são do tipo **normal de pesos 2 e 1**, respectivamente. Seus graus de semelhança são valores pré-definidos e as combinações possíveis estão dispostas nas Tabelas 8 e 9.

Tabela 8 – Grau de semelhança para o atributo Porte.

	Pequeno	Médio	Grande
Pequeno	100%	50%	0%
Médio	50%	100%	50%
Grande	0%	50%	100%

Fonte: Próprios autores (2015).

Tabela 9 – Grau de semelhança para o atributo Pelagem.

	Curto	Longo
Curto	100%	50%
Longo	50%	100%

Fonte: Próprios autores (2015).

Por fim, a **raça** é um atributo do tipo **bônus** de **peso 3**. O grau de semelhança também é de apenas 0% ou 100%. O cálculo de semelhança para este atributo é simples: raças iguais, grau de 100%, caso contrário, 0%. Não deseja-se que este atributo afete negativamente a semelhança entre anúncios, devido à facilidade de um usuário leigo informá-lo incorretamente.

A Tabela 10 agrupa todos os atributos avaliados com seus respectivos tipos e pesos.

Tabela 10 – Atributos comparados, seus tipos e pesos.

Nome	Tipo	Peso
Conjunto de cores	Exclusivo	3
Gênero	Exclusivo	3
Nome	Exclusivo	3
Momento	Exclusivo	2
Porte	Normal	2
Local	Normal	2
Pelagem	Normal	1
Peculiaridades	Bônus	Variável
Raça	Bônus	3

Fonte: Próprios autores (2015).

4.2 SEMELHANÇA RESULTANTE

A comparação entre anúncios resulta em um grau de 0% a 100%. Esse grau representa o quão semelhantes são os anúncios. Para calcular o grau de semelhança de dois anúncios, são utilizados os atributos listados anteriormente, bem como seus tipos e pesos.

Os primeiros atributos a serem comparados são os exclusivos. Como dito anteriormente, se a comparação entre valores de qualquer atributo do tipo exclusivo resultar em grau de semelhança 0%, o algoritmo para o grau de semelhança resultante é 0%. Caso contrário, os graus de semelhança calculados e seus respectivos pesos são reservados para o cálculo final. Após o processamento do tipo exclusivo, os atributos normais também têm seus graus calculados e reservados com os pesos.

Processados os atributos exclusivos e normais, uma média ponderada parcial (MP) entre graus e pesos é calculada, usando a Equação 4.2. Assumindo os atributos A e B da Tabela 11 como exemplo, fazendo parte da MP, o somatório dos graus de semelhança multiplicados pelos pesos ($\sum GS \times P$) é 300% e dos pesos ($\sum P$) é 4. Aplicando a fórmula da Equação 4.2, a MP resulta em 75%.

$$MP = \frac{\sum(GS \times P)}{\sum P} \quad (4.2)$$

Tabela 11 – Exemplo de atributos exclusivos e normais

Atributo	Tipo	Peso	Grau de semelhança (%)
A	Exclusivo	3	100
B	Normal	1	0

Fonte: Próprios autores (2015).

Por fim, o sistema processa os atributos bônus, calculando seus graus de semelhança e reservando-os, juntamente com seus pesos. Após calcular os graus, o sistema descarta os que são menores ou iguais à média ponderada parcial, de modo que sobrem apenas os que contribuem positivamente para a semelhança resultante. Cada atributo bônus, em ordem crescente de grau de semelhança, fará parte de uma nova média ponderada parcial (MP_{nova}) com a MP anterior e seu somatório de pesos ($\sum P_{MP}$). Considerando o grau de semelhança ($GS_{bônus}$) e o peso ($P_{bônus}$) do atributo bônus, o cálculo de cada nova MP é dado pela Equação 4.3.

$$MP_{nova} = \frac{(MP \times \sum P_{MP}) + (GS_{bônus} \times P_{bônus})}{\sum P_{MP} + P_{bônus}} \quad (4.3)$$

Seguindo o exemplo da Tabela 11, se considerarmos os atributos bônus da Tabela 12, temos os seguintes resultados:

1. o atributo Z não é utilizado, pois seu grau de semelhança (70%) é menor do que a média ponderada parcial (75%).
2. o primeiro atributo a ser utilizado é o Y, que possui o menor grau de semelhança dentre os atributos a serem considerados. Aplicando a formula da Equação 4.3, a nova média ponderada parcial é 78% e o novo somatório de pesos, 5;
3. em seguida, é utilizado o atributo X. Aplicando-se a fórmula novamente, a nova média é 84,29% e o novo somatório de pesos, 7.

Tabela 12 – Exemplo de atributos Bô-nus

Nome	Peso	Grau de semelhança (%)
X	2	100
Y	1	90
Z	3	70

Fonte: Próprios autores (2015).

A última média ponderada parcial calculada, por fim, é o grau de semelhança resultante da comparação entre dois anúncios. No exemplo utilizado, o grau resultante é 84,29%.

5 IMPLEMENTAÇÃO

O sistema Bicho Perdido foi estruturado em diversas camadas, de modo a organizar melhor os seus componentes, facilitando sua construção, bem como sua futura ampliação e manutenção. As macrocamadas **Cliente** e **Serviço** e suas subcamadas são apresentadas na Figura 26 e a seguir:

Cliente Macrocamada que contém a interface gráfica do usuário (GUI¹), dividida em módulos por funcionalidade, e também contém o seguinte componente:

Interceptor Componente à parte, comum a todos os módulos, que trata todas as requisições HTTP feitas à camada Serviço.

Serviço Macrocamada que contém uma interface de comunicação, a lógica de negócio e o armazenamento permanente dos dados. É composta pelas seguintes subcamadas:

Fachada Camada que implementa a interface de comunicação com a GUI. Neste caso, a camada foi implementada como um *web service* RESTful.

Negócio É onde reside a lógica de negócio: validações e tratamentos de dados. Todo o fluxo de dados é tratado por esta camada antes de ser direcionado para outra.

Persistência Camada responsável por comunicar-se com o SGBD, persistindo ou carregando os dados necessários pelo serviço.

Assíncrona Executa pequenos serviços independentes em *threads* separadas. Os serviços assíncronos executados são: remoção de *tokens* de autenticação vencidos, envio de notificações e cálculo de semelhança. Esses serviços serão detalhados adiante.

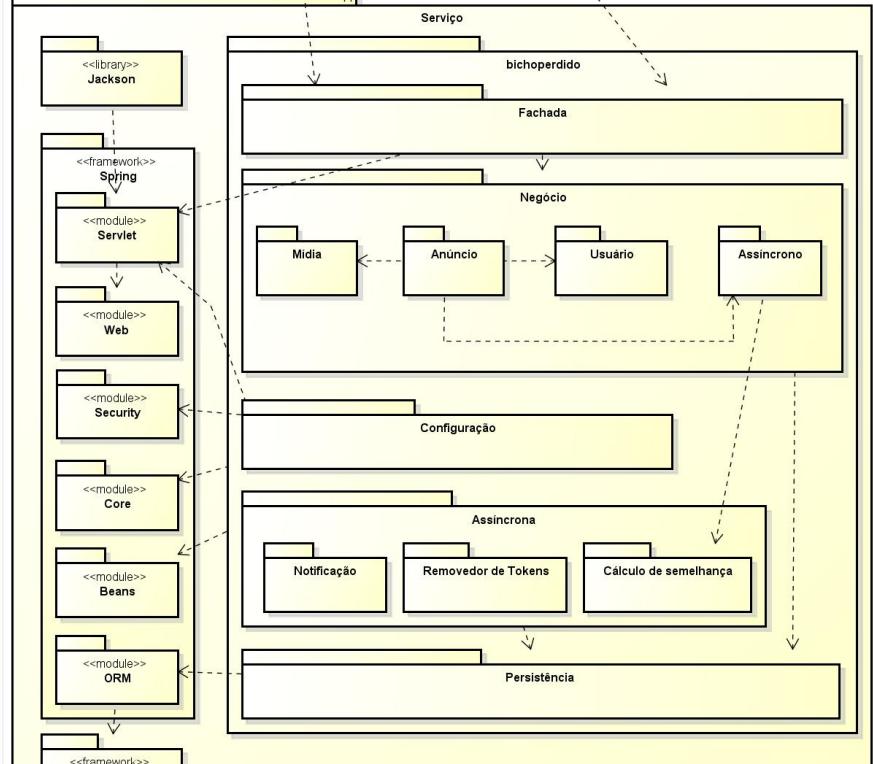
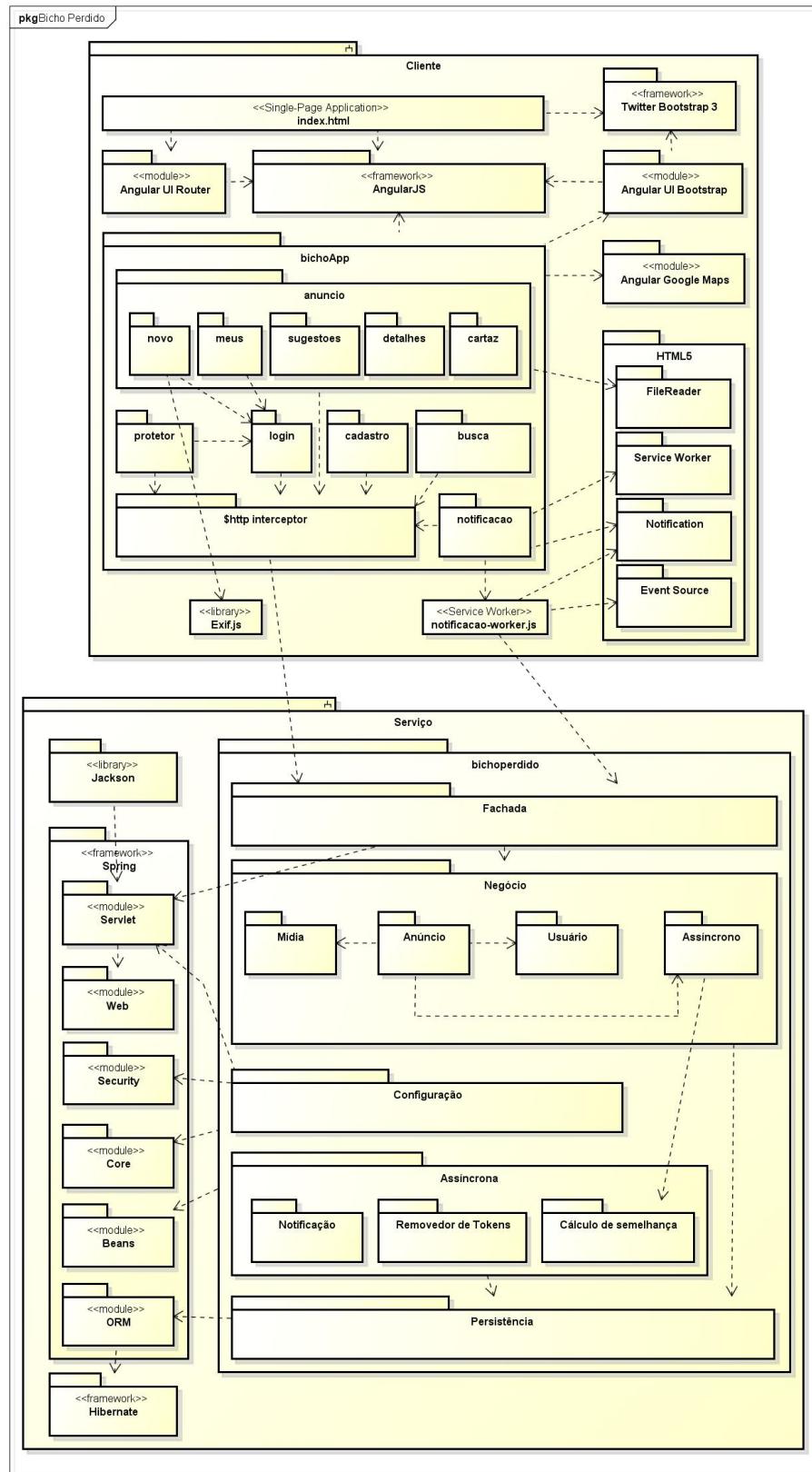
Configuração Nesta camada encontram se todos os componentes que afetam o serviço inteiro, tais como *Servlet Filters*, captura e conversão de exceções para códigos de erro e os parâmetros de configuração do serviço. Os *Servlet Filters* serão explicados mais adiante.

5.1 CLIENTE

A macrocamada Cliente contém o código da interface gráfica do usuário (GUI), bem como *frameworks* e outros componentes utilizados.

¹ Acrônimo de *graphical user interface*, termo em inglês de mesmo significado.

Figura 26 – Diagrama de pacotes.



Fonte: Próprios autores (2015).

5.1.1 FRAMEWORKS UTILIZADOS

A GUI do sistema foi construída no topo do *framework* arquitetural AngularJS², do Google. O AngularJS é baseado na linguagem JavaScript e oferece características interessantes para a construção rápida de aplicações para a *web*, como as que seguem:

- extensão do vocabulário HTML. Esta característica contempla diversas necessidades das páginas dinâmicas, como associação de dados variáveis a formulários e outros componentes visuais (*two-way data binding*), estruturas condicionais, de repetição, entre outras.
- organização do código em módulos, para estruturação baseada em funcionalidades ou criação de componentes reutilizáveis.
- serviços que facilitam a comunicação com *web services*.

Com o auxílio do AngularJS, a GUI do Bicho Perdido foi desenvolvida usando o conceito de *single-page application* (SPA). As SPAs assemelham-se às *apps* móveis nativas e aplicações *desktop*, pois tratam-se de uma única página HTML controlando todo o fluxo. Durante a execução de uma SPA, nenhuma requisição de atualização de página completa é feita pelo navegador e todo conteúdo dinâmico é atualizado pela manipulação do documento pela linguagem JavaScript – usando os conceitos do *Document Object Model* (DOM) – o que oferece ao usuário uma experiência de uso mais fluída. Normalmente as SPAs recebem conteúdo de *web services*, como no caso deste sistema.

As telas do sistema foram baseadas nos componentes visuais oferecidos pelo *framework* de apresentação Twitter Bootstrap³, que tem foco em projetos responsivos, ou seja, cuja apresentação se adapta ao dispositivo utilizado – mais especificamente, ao tamanho da tela – visando as plataformas móveis, em especial o sistema operacional Android⁴, do Google.

5.1.2 TIPOS DE APLICATIVOS MÓVEIS

Existem atualmente três abordagens possíveis de construção e distribuição de aplicativos móveis (*apps*): *web*, nativa e híbrida.

Na primeira abordagem, a construção usa as mesmas tecnologias abertas das páginas da *web* – HTML, CSS⁵ e JavaScript – e a distribuição é feita através do navegador. A vantagem desta abordagem é a possibilidade de reaproveitamento de habilidades em

² <<http://www.angularjs.org/>>.

³ <<http://www.getbootstrap.com>>.

⁴ <<http://www.android.com/>>.

⁵ *Cascading Style Sheets*, ou folhas de estilo em cascata (tradução nossa).

desenvolvimento para *web* e a manutenção de um único código-fonte para diversas plataformas, em teoria. O código-fonte único só é possível na prática quando a implementação das especificações das tecnologias utilizadas for padronizada nas plataformas alvo, o que não acontece atualmente em suas versões mais recentes. A desvantagem fica por conta do desempenho, pois o código roda sobre mais uma camada de *software*, além do sistema operacional.

Na segunda, a construção é realizada em ambiente de desenvolvimento específico, na linguagem nativa suportada pelo dispositivo – Java, no Android, por exemplo – e a distribuição é via pacote de instalação, se suportado, ou loja de aplicativos, onde é necessário ser previamente cadastrado como desenvolvedor e distribuidor de *apps*, podendo haver custos. A vantagem desta abordagem, além do desempenho e do aproveitamento pleno das capacidades não acessíveis pelo navegador, está na possibilidade de divulgar os *apps* a custos acessíveis, bem como vendê-los, usando as lojas. A desvantagem está nos custos de distribuição – caso o objetivo não seja compensá-los – e na necessidade de aprovação do *software* antes da publicação, bem como todas as suas versões subsequentes. Também não é garantido que a versão em uso no cliente seja sempre a atual, sem que o desenvolvedor codifique a desativação de versões não suportadas.

A abordagem híbrida é uma opção mais recente, que combina o uso das tecnologias da *web* na construção com a distribuição como *apps* nativas. Esta abordagem envolve o uso de algum *framework* – como o Cordova da Apache, PhoneGap da Adobe, Ionic (código aberto), Supersonic da AppGyver, por exemplo – que nada mais são que *apps* nativos sem nenhuma tela, com a capacidade de exibir páginas *web* e até mesmo componentes nativos – botões, barras de título, etc. – desde que o desenvolvedor utilize as chamadas específicas do *framework* para isso. A vantagem está na teórica possibilidade de manter código-fonte único para plataformas diferentes e no ganho de desempenho e capacidades – como as não acessíveis atualmente pelo navegador, como o sensor de movimento, por exemplo – desde que suportados pelo *framework*. A desvantagem é que a solução híbrida não é tão performática quanto a nativa e requer o aprendizado dos *frameworks*.

Com o foco na facilidade de manutenção e distribuição do sistema Bicho Perdido, decidiu-se pela primeira abordagem (*web*). Caso algum requisito não pudesse ser entregue pela existência de limitações das tecnologias atuais, o plano de contingência seria a adoção de um *framework* híbrido, o que não foi necessário.

5.1.3 MÓDULOS

Quanto à organização do código-fonte da camada Cliente, todo código que roda sobre o AngularJS está dividido em módulos, cada módulo representando uma funcionalidade esperada, facilitando a rastreabilidade dos requisitos. Um módulo em AngularJS é um conjunto de diferentes componentes, reutilizáveis ou não, divididos nas seguintes categorias:

Diretivas São marcações em elementos do DOM (*tags*, atributos, classes CSS ou comentários) que adicionam comportamentos ou transformam estes elementos e seus descendentes. Por exemplo, o atributo `ng-model` associa um controle de formulário HTML a uma variável de escopo. O *framework* traz um conjunto de diretivas úteis pré-definidas e permite ao desenvolvedor criar as suas próprias.

Controladores São funções construtoras associadas ao HTML, inteiro ou a um trecho, através da diretiva `ng-controller`, que possuem duas funções básicas: inicializar as variáveis de seu escopo e adicionar comportamento a elas. Em uma analogia com o padrão arquitetural MVC (*model-view-controller*), o HTML seria a *view*, o escopo seria o *model* e o controlador seria o *controller* e conteria a lógica entre os dois.

Serviços São porções substituíveis de código JavaScript, utilizadas em conjunto em outros componentes através de injeção de dependência. Os serviços são criados para organizar e compartilhar código para toda a aplicação, são instanciados pelo AngularJS somente quando necessários (*lazy*) e uma única vez (*singleton*). O *framework* oferece um conjunto de serviços úteis, como o `$http` para interação com *web services*, além de permitir a criação de novos serviços pelo desenvolvedor.

Filtros Formatam valores para serem exibidos ao usuário e podem ser usados em *views*, controladores ou serviços.

Os módulos do sistema Bicho Perdido são: `anuncio` – composto pelos submódulos `novo`, `meus`, `sugestoes`, `detalhes` e `cartaz` – `protetor`, `login`, `cadastro`, `busca` e `notificacao` e estão ilustrados pela Figura 26.

5.1.4 *INTERCEPTOR*

Um componente à parte da camada cliente, comum a todos os módulos, é o interceptador de requisições HTTP, ou *interceptor*. O *interceptor* realiza três funções na aplicação:

- caso o usuário ativo do sistema esteja autenticado, adiciona o cabeçalho `x-token` a todas as requisições HTTP feitas ao serviço, como parte da estratégia de integração da segurança, detalhada na seção 5.3;
- trata todas as respostas com erros conhecidos do serviço, identificadas por códigos de erro específicos da aplicação. Exceto em caso de autenticação expirada ou inválida no serviço, o *interceptor* é responsável por exibir alertas de erro com as mensagens adequadas. Mais detalhes sobre segurança e tratamento de erros encontram-se na seção 5.3;

- trata os erros desconhecidos, exibindo alertas com sentido na GUI, permitindo a descoberta e rastreamento de possíveis falhas no sistema, após ter entrado em produção.

5.1.5 LOCAL DAS FOTOS

A identificação do local (coordenadas geográficas – latitude e longitude) onde uma fotografia digital foi capturada, utilizando os metadados do arquivo de imagem – funcionalidade prevista no projeto do Bicho Perdido – foi implementada integralmente na camada Cliente, utilizando a API FileReader do HTML5 e a biblioteca JavaScript Exif.js⁶.

A API FileReader permite que aplicações *web* leiam, de maneira assíncrona, o conteúdo de arquivos no dispositivo do usuário. Essa API viabiliza, por exemplo, que uma imagem seja exibida na página da *web* tão logo seja escolhida pelo usuário dentre seus arquivos de imagem em disco, sem necessidade de processamento no servidor.

A biblioteca Exif.js, de código aberto, faz a leitura dos dados binários do arquivo de imagem, localiza os metadados e os interpreta *bit a bit*, de acordo com a especificação EXIF⁷, versão 2.2, permitindo acesso às informações em alto nível. EXIF é uma especificação seguida por diversos fabricantes de câmeras digitais, compatível com os formatos de imagem JPEG⁸ e TIFF⁹ e armazena dados relativos à captura, como data e hora, latitude e longitude, etc. (JEITA, 2002).

A funcionalidade foi testada nos dispositivos móveis Motorola Moto G segunda geração com Android versão 5.0.2 e Apple iPhone 5C com iOS versão 9.0.2 sem sucesso, mas opera normalmente no Google Chrome versão *desktop*. Nos testes nesses dispositivos móveis, descobriu-se que várias informações do EXIF são suprimidas quando o arquivo de imagem é lido pela API FileReader, inclusive a latitude e a longitude. Acredita-se que o comportamento faça parte de uma política de segurança dos dispositivos. Porém, se em alguma versão futura dos sistemas operacionais mencionados for possível permitir, via configuração do usuário por exemplo, a leitura completa do EXIF por aplicações *web*, a funcionalidade deverá operar normalmente.

5.2 SERVIÇO

É a macrocamada responsável por prover os dados que a GUI precisa, tratar dados de entrada e saída, bem como armazenar e carregar informações do sistema. O Serviço foi desenvolvido em linguagem Java e em grande parte estruturado pelo *framework*

⁶ Disponível em <<https://github.com/exif-js/exif-js>>. Acesso em: nov. 2015.

⁷ *Exchangeable Image File Format*, ou formato de arquivo de imagem intercambiável (tradução nossa).

⁸ Formalmente conhecido como JFIF (*JPEG File Interchange Format*). JPEG é acrônimo de *Joint Photographic Experts Group*, criadores do formato.

⁹ Acrônimo de *Tagged Image File Format*.

Spring¹⁰. Spring é um *framework* vertical, separado em diversos módulos que auxiliam o desenvolvimento de diferentes camadas de uma aplicação. Os módulos do Spring utilizados no Serviço são:

Core e Beans São os módulos base do *framework* e provêm inversão de controle (IoC, acrônimo de *Inversion of Control*, em inglês) e injeção de dependência (DI, acrônimo de *Dependency Injection*, em inglês). Inversão de controle é um padrão de projeto no qual os métodos de um objeto de uma determinada classe não criam instâncias de outras classes. A injeção de dependência, que resolve a inversão de controle, consiste em passarmos objetos necessários aos métodos como parâmetros. Uma aplicação desenvolvida sobre o Spring delega a criação de objetos totalmente para o *framework* e apenas os utiliza. Com isso é possível reduzir o acoplamento entre as classes, criar classes mais específicas, consequentemente reutilizáveis, facilitando o crescimento do código e diminuindo o efeito colateral das alterações no código de seus dependentes.

ORM Este módulo fornece uma camada de integração (IoC/DI) do Spring com mapeadores objeto-relacional, que fazem a correspondência entre entidades armazenadas em bancos de dados relacionais com objetos Java. Entre os mapeadores, estão o *framework* de persistência Hibernate¹¹, bem como as APIs JPA¹² e JDO¹³.

Web e Servlet Estes módulos fornecem uma camada de abstração do protocolo HTTP, facilitando a manipulação do cabeçalho e do corpo das requisições recebidas e respostas de uma aplicação. Através de anotações específicas do Spring no código Java, é possível mapear facilmente URIs de acesso para métodos, bem como configurar o tipo dos dados de entrada e saída (JSON, XML, HTML, etc).

Security Módulo que provê autenticação e autorização para aplicações Java. Permite associar níveis de acesso a credenciais de usuário e definir os níveis necessários para cada URI de uma aplicação. Também controla o acesso, liberando ou bloqueando com base nas configurações efetuadas. Por fim, disponibiliza automaticamente para a aplicação informações sobre o usuário corrente, através de DI.

A Fachada, subcamada do Serviço, foi construída utilizando-se os módulos *Web* e *Servlet* do Spring, em conjunto com a biblioteca Jackson¹⁴, que faz o mapeamento automático entre objetos JSON e classes Java. A camada foi separada em quatro *REST Controllers*, conforme as funcionalidades pelas quais são responsáveis, para facilitar a manutenção. Um *REST Controller* no Spring é uma classe responsável por um ou mais

¹⁰ <<https://spring.io>>.

¹¹ <<http://hibernate.org>>.

¹² Acrônimo de *Java Persistence API*.

¹³ Acrônimo de *Java Data Objects*.

¹⁴ <<https://github.com/FasterXML/jackson>>.

mapeamentos de métodos Java para URIs (ou pontos de acesso) do *web service*. Os *REST Controllers* do Bicho Perdido são:

Mídia Responsável por requisições que envolvam a manipulação – recepção, armazenamento, envio e remoção – de arquivos de mídia, como imagens e vídeos.

Anúncio Responsável por requisições que envolvam armazenamento, recuperação, remoção e edição de anúncios de animais perdidos e encontrados, bem como envio das listas de possíveis raças ou peculiaridades (informações armazenadas no banco de dados).

Usuário Responsável por requisições que envolvam os usuários do sistema, como cadastro, autenticação, dados de protetor e envio de informações sobre o usuário corrente.

Assíncrono Responsável por requisições relacionadas com os serviços assíncronos, como inscrição para o recebimento de notificações e recuperação de anúncios semelhantes, previamente calculados pela camada Assíncrona.

A camada Configuração especifica os parâmetros de conexão com o banco de dados, configurações para uso do HTTPS, configurações de registro de eventos (*log*) e contém os *Servlet Filters*. Os *Servlet Filters* são componentes que o Spring injeta em seu módulo *Servlet* e que agem como filtros modificadores para todas as requisições e respostas HTTP, entre a GUI e a Fachada do Serviço. O sistema implementa dois *Servlet Filters*:

- um deles é responsável pela implementação do *Cross-Origin Resource Sharing*¹⁵ (CORS). O CORS consiste basicamente em cabeçalhos HTTP que especificam quais origens (URIs de onde partem as requisições) estão habilitadas a terem suas requisições HTTP respondidas pelo *web service*. Este *Service Filter* configura permissão a qualquer origem, tornando a camada Serviço independente e funcional para qualquer implementação de GUI em diferentes locais da *web*. Todos os navegadores da *web* implementam, por segurança, a *Same-Origin Policy* (política de mesma origem), permitindo que apenas recursos “locais” (com relação à quem inicia a requisição) sejam acessados. O CORS é a recomendação atual da W3C para compartilhar recursos entre origens diferentes;
- o outro é responsável pela identificação e tratamento do cabeçalho *x-token* nas requisições de entrada. O *token* contido no cabeçalho é a identificação exclusiva de qual usuário está autenticado e em qual instância da GUI. Este *Service Filter* usa o *token* para identificar o usuário corrente e informar ao módulo *Security* do Spring, que distribui a informação para toda a aplicação. Além disso, requisita ao Serviço a prorrogação da validade do *token* recebido sempre que este for válido.

¹⁵ Compartilhamento de recursos entre origens distintas (tradução nossa).

A camada Negócio, única acessada pela Fachada, foi separada em quatro pacotes (ou componentes), com os mesmos nomes e escopos dos *REST Controllers* existentes na Fachada. Esses pacotes contêm as classes necessárias para validações, conversões e outros tratamentos de dados. Esta camada comunica-se ainda com as camadas Persistência e Assíncrona.

A camada Persistência é responsável pelo armazenamento e recuperação de informações do banco de dados e foi desenvolvida sobre o módulo ORM do Spring e sobre o *framework* de persistência Hibernate. Esta camada é dividida em onze partes, cada uma representando uma entidade do modelo ER (entidade-relacionamento) do sistema. As entidades, bem como seus relacionamentos, estão ilustradas no pacote `persistence` do diagrama de classes da Figura 27 e seguem o mesmo modelo do banco de dados.

A maior parte das requisições da GUI para o Serviço segue o mesmo modelo: a Fachada recebe os dados, passa-os para o componente responsável na camada Negócio, que trata e valida esses dados e, por sua vez, delega para a camada Persistência o armazenamento das informações. Especificamente na criação de um novo anúncio, a camada Negócio solicita uma operação adicional à Persistência: calcular imediatamente a distância geográfica entre o local (latitude e longitude) da ocorrência registrada no novo anúncio e os locais de todos os anúncios criados antes. A camada Persistência, por sua vez, delega a operação para o banco de dados, usando diretamente a linguagem SQL¹⁶. O SGBD utilizado, o PostgreSQL¹⁷, traz um módulo adicional chamado `earthdistance` que calcula de forma rápida e otimizada essas distâncias geográficas, usando a matemática do grande círculo.

A camada Assíncrona comunica-se diretamente com a Persistência e executa três serviços em paralelo com a aplicação principal, cada um em um processo leve (*thread*) independente. Os três serviços são:

- o **removedor de tokens** de autenticação com a validade expirada, para que não se acumulem indefinidamente no banco de dados;
- o gerente de **notificações**, que recebe requisições do Serviço para enviar mensagens JSON (ou texto) ao dono de um *token* específico. Este serviço enfileira as notificações internamente, verifica constantemente essa fila e, se há notificações não enviadas, envia-nas ao Cliente destinatário, de acordo com o *token* indicado. No caso de erro, a notificação não enviada entra na fila novamente, por quatro erros consecutivos, até ser descartada. O protocolo de envio é descrito na seção 5.3;
- o serviço de **cálculo de semelhança** entre anúncios cadastrados no sistema, detalhado no diagrama de classes da Figura 27, é dividido nos seguintes componentes:

¹⁶ Structured Query Language, ou linguagem de consulta estruturada (tradução nossa).

¹⁷ <<http://www.postgresql.org/>>.

Comparadores de atributo Estes componentes são responsáveis pelo cálculo da semelhança de cada atributo de dois anúncios diferentes. Podem ser dos tipos exclusivo, normal ou bônus – de acordo com o tipo do atributo – recebem como entrada os valores do atributo dos anúncios que estão sendo comparados e retornam como saída o peso e o grau de semelhança. Cada comparador possui uma implementação específica, de acordo com o algoritmo descrito no Capítulo 4. Porém, visando maior desempenho e menor processamento no Serviço, todos os atributos exclusivos, exceto o conjunto de cores do animal, são comparados pelo SGBD através de SQL.

Comparador de anúncio Este componente é responsável pelo cálculo da semelhança resultante entre dois anúncios. Recebe como entrada uma lista ordenada de comparadores de atributo e, seguindo a ordem, executa cada um. A implementação do comparador de anúncio segue a especificação do algoritmo descrito no Capítulo 4.

Carregador de anúncios Este componente é responsável por carregar do banco de dados, para cada anúncio de origem, os outros anúncios que serão comparados com ele. Possui duas implementações:

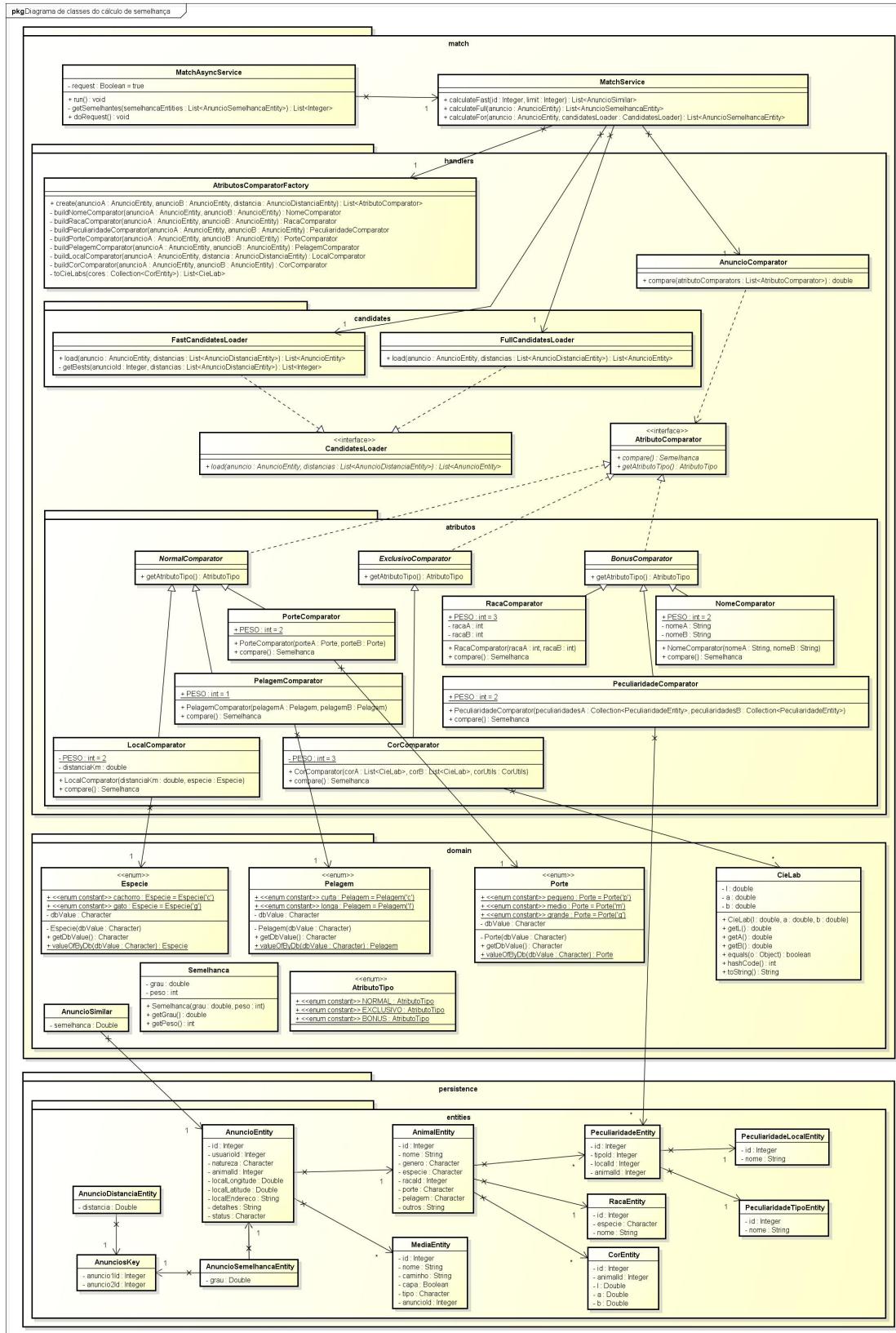
- uma delas carrega todos os anúncios que ainda não foram comparados com o anúncio de origem;
- a outra carrega apenas os cem anúncios mais próximos ao de origem, considerando o local da ocorrência. Esta implementação é uma estratégia de otimização que visa atender à funcionalidade de sugerir anúncios semelhantes a um novo anúncio, tão logo ele seja cadastrado pelo usuário no sistema, visto que a complexidade do algoritmo de recomendação, em notação assintótica, é $O(n^2)$, ou seja, seu processamento cresce proporcional ao quadrado da quantidade de anúncios cadastrados.

Fábrica de comparadores de atributos Este componente é responsável por criar a lista de comparadores de atributo, na ordem que devem ser executados pelo comparador de anúncio, de acordo com o algoritmo descrito no Capítulo 4.

5.3 INTEGRAÇÃO

Esta seção relata a experiência de integração das camadas Cliente e Serviço. Ambas camadas foram desenvolvidas de forma independente na maior parte do tempo com o menor acoplamento possível. A comunicação entre as camadas acontece exclusivamente por requisições HTTP, com a transmissão de formatos leves, como JSON e texto puro. O baixo acoplamento permite a substituição de uma camada inteira, sem provocar mudanças no

Figura 27 – Diagrama de classes do microserviço de cálculo de semelhança.



Fonte: Próprios autores (2015).

código da outra, apenas mantendo o formato de intercâmbio de dados, que é independente de plataforma e linguagem de programação.

5.3.1 TRATAMENTO DE ERROS

Erros detectados pelo Serviço após uma requisição, de validação ou imprevistos, são informados na resposta da requisição em um objeto JSON, que contém um valor numérico padronizado associado a uma chave de nome `bpError`. Essa estratégia permite que as mensagens de erro possam ser customizadas na GUI, desde que mantido o significado. Os significados dos códigos de erro numéricos do Bicho Perdido estão listados na Tabela 13.

Tabela 13 – Tabela de códigos de erro do sistema.

Código numérico	Significado do erro
1	Credenciais de autenticação inválidas (<i>e-mail</i> e senha).
2	<i>Token</i> necessário e não enviado ou inválido.
3	Endereço de <i>e-mail</i> já cadastrado anteriormente.
4 a 14	Valor inválido no cadastro de um anúncio.
15 a 19	Operação permitida somente ao dono do anúncio.
20 a 24	Valor inválido no cadastro de usuário.
25 e 26	Valor inválido nos dados do protetor.
27	Anúncio resolvido não pode ser alterado.
28 a 35	Filtros de busca inválidos.
98	Erro interceptado porém não previsto pelo Serviço.
99	Erro desconhecido.

Fonte: Próprios autores (2015).

O erro 99 é apenas conceitual. Seu significado é de que houve um erro no Serviço, indicado por um código de erro HTTP na resposta da requisição, que não trouxe um objeto JSON com a chave `bpError`. Na GUI do Bicho Perdido, o *interceptor* é responsável por identificar a presença da chave `bpError` na resposta das requisições e exibir a mensagem mais adequada em um alerta, exceto no caso do erro 2, onde é exibida a janela de autenticação.

5.3.2 AUTENTICAÇÃO

Algumas funcionalidades da aplicação devem ser acessadas somente por usuários autenticados. Para que isso funcione, as camadas Cliente e Serviço implementam componentes de autenticação e precisam estar em constante sincronia. Essa sincronia ocorre em três momentos distintos:

- quando o usuário solicita sua autenticação fornecendo suas credenciais de acesso (*e-mail* e senha). A sincronia ocorre da seguinte forma:

1. o Cliente envia as credenciais do usuário para o Serviço em uma requisição HTTP;
 2. o Serviço verifica se as credenciais estão corretas;
 3. se estiverem, o Serviço gera um *token* que identifica unicamente o usuário e a instância da GUI utilizada, define sua validade para sessenta minutos e o retorna na resposta da requisição;
 4. o Cliente armazena localmente o *token* recebido e passa a enviá-lo ao serviço no cabeçalho de todas as requisições subsequentes.
- quando o usuário faz qualquer requisição a uma URI protegida no Serviço com um *token* inválido no cabeçalho. Nesse momento, a sincronia ocorre da seguinte forma:
 1. o Cliente inicia a requisição;
 2. o Serviço verifica se o *token* está inválido;
 3. se estiver, o Serviço retorna um objeto JSON com o valor 2 na chave `bpError`;
 4. o Cliente remove a cópia do *token* armazenada localmente e solicita que as credenciais do usuário sejam informadas na janela de autenticação.
 - quando o usuário solicita a saída (*log out*) do sistema. Nesse momento, a sincronia ocorre da seguinte forma:
 1. o Cliente remove a cópia do *token* armazenada localmente;
 2. o Cliente inicia uma requisição para a URI de *log out* do Serviço;
 3. o Serviço remove o *token*, informado no cabeçalho da requisição, do banco de dados.

5.3.3 NOTIFICAÇÕES

Sempre que um anúncio for cadastrado no sistema, se o local da ocorrência estiver dentro da área de cobertura de um usuário protetor, este será notificado sobre a existência desse anúncio. Além disso, após comparar a semelhança do novo anúncio (origem) com os demais (alvos), sempre que a comparação resultar em grau 50% ou maior, o dono do anúncio alvo é notificado sobre a existência do anúncio origem.

As notificações foram implementadas utilizando três APIs do HTML5: *Notification*, *Server-Sent Events* (SSE) e *Service Workers*.

A *Notification API* permite que uma aplicação *web* emita notificações ao seu usuário. A API SSE permite que uma aplicação *web* receba dados (eventos do servidor, em formato texto) de um *web service* através de uma conexão HTTP persistente. Por fim, os *Service Workers* são códigos em JavaScript de uma aplicação *web* que executam

de forma paralela e independente da página sendo exibida, comunicando-se com esta por troca de mensagens totalmente assíncronas. No Bicho Perdido, o *Service Worker* inicia e interrompe uma conexão SSE de acordo com o estado do usuário: autenticado ou não.

O fluxo de funcionamento das notificações no Bicho Perdido consiste nos seguintes passos:

1. a camada Cliente verifica o suporte aos *Service Workers* no navegador utilizado;
2. caso sejam suportados, o Cliente verifica o suporte à *Notification API*;
3. caso seja suportada, o Cliente solicita a permissão do usuário para emitir notificações ou verifica se já há permissão;
4. caso tenha permissão, o Cliente inicia a execução do *Service Worker* responsável pelas notificações;
5. inicialmente, o *Service Worker* fica apenas esperando receber alguma mensagem da página principal;
6. quando o usuário do sistema se autentica com sucesso, a página principal informa o *token* armazenado ao *Service Worker*;
7. o *Service Worker* inicia uma conexão SSE com o Serviço, com o *token* como parâmetro (na própria URI) e fica aguardando o recebimento de eventos do servidor (mensagens SSE);
8. quando um dos eventos geradores de notificações ocorre no Serviço, este emite o tipo de notificação a ser mostrado e outros dados associados em um objeto JSON como uma mensagem SSE para as conexões correspondentes aos *tokens* destinatários;
9. o *Service Worker* recebe a mensagem SSE e emite a notificação usando a *Notification API*.

6 TESTE DE DESEMPENHO

Para analisar o desempenho do algoritmo de recomendação, foram realizados diversos testes, nos quais informações foram inseridas automaticamente no banco de dados através da camada Serviço. A partir de registros de eventos (*logs*) gerados pelo algoritmo, foram feitas análises e gráficos para ilustrar seu desempenho, bem como estimá-lo para populações maiores. Os critérios utilizados para popular o sistema, assim como as análises realizadas, seus resultados e gráficos, serão descritos a seguir.

6.1 POPULAÇÃO

Os critérios utilizados para a população foram selecionados visando um meio termo entre um cenário controlado para o teste e um cenário real. São eles:

- distribuição aleatória de anúncios de natureza perdido e encontrado;
- momento (data e hora) aleatório no ano de 2015;
- local da ocorrência aleatório limitado a uma região retangular abrangendo a cidade de Porto Alegre, Rio Grande do Sul (da latitude $-30,092$ à $-30,003$ e da longitude $-51,208$ à $-51,088$);
- distribuição aleatória de espécies cachorro e gato;
- distribuição aleatória de gêneros macho e fêmea para perdidos e também do valor “não sei” para encontrados;
- nomes aleatórios para anúncios de perdidos somente, sem influência no algoritmo;
- quantidade de cores no conjunto aleatória, de uma a três, sendo a primeira (ou única) aleatória entre preto (#000000) e branco (#FFFFFF) e as demais, aleatórias entre preto, branco, marrom (#A52A2A), chocolate (#D2291E), cinza (#A9A9A9) e cinza escuro (#333333), sem repetir;
- raça aleatória, considerando as raças possíveis para a espécie;
- porte aleatório para cachorros, entre pequeno, médio e grande;
- pelagem aleatória, entre curta e longa;
- quantidade aleatória de peculiaridades, de nenhuma a três, de tipo e local aleatórios dentre os possíveis valores.

6.2 TESTES REALIZADOS

Para determinar o desempenho, bem como projetá-lo considerando o aumento da quantidade de anúncios cadastrados, foram realizados cinco testes: com mil, dois mil, três mil, quatro mil e cinco mil anúncios criados, cada um vinculado a um usuário diferente no sistema. As Tabelas 14, 15, 16, 17 e 18, respectivamente, descrevem a distribuição das amostras utilizadas em cada teste.

Tabela 14 – Distribuição da amostra do primeiro teste.

Natureza	Espécie	Gênero	Quantidade
encontrado	cachorro	fêmea	71
encontrado	cachorro	macho	88
encontrado	cachorro	não sei	87
encontrado	gato	fêmea	90
encontrado	gato	macho	86
encontrado	gato	não sei	94
perdido	cachorro	fêmea	120
perdido	cachorro	macho	134
perdido	gato	fêmea	118
perdido	gato	macho	112

Fonte: Próprios autores (2015).

Tabela 15 – Distribuição da amostra do segundo teste.

Natureza	Espécie	Gênero	Quantidade
encontrado	cachorro	fêmea	176
encontrado	cachorro	macho	172
encontrado	cachorro	não sei	150
encontrado	gato	fêmea	171
encontrado	gato	macho	154
encontrado	gato	não sei	165
perdido	cachorro	fêmea	246
perdido	cachorro	macho	253
perdido	gato	fêmea	260
perdido	gato	macho	253

Fonte: Próprios autores (2015).

A partir dos *logs* dos testes, foram coletadas as seguintes estatísticas, demonstradas na Tabela 19:

- o tempo total de comparação dos atributos exclusivos realizada pelo SGBD, como parte da estratégia de otimização do algoritmo;
- o tempo total de comparação efetuado pela implementação do algoritmo em linguagem Java dos anúncios não filtrados pelo SGBD;

Tabela 16 – Distribuição da amostra do terceiro teste.

Natureza	Espécie	Gênero	Quantidade
encontrado	cachorro	fêmea	248
encontrado	cachorro	macho	236
encontrado	cachorro	não sei	253
encontrado	gato	fêmea	232
encontrado	gato	macho	265
encontrado	gato	não sei	244
perdido	cachorro	fêmea	388
perdido	cachorro	macho	384
perdido	gato	fêmea	393
perdido	gato	macho	357

Fonte: Próprios autores (2015).

Tabela 17 – Distribuição da amostra do quarto teste.

Natureza	Espécie	Gênero	Quantidade
encontrado	cachorro	fêmea	347
encontrado	cachorro	macho	314
encontrado	cachorro	não sei	318
encontrado	gato	fêmea	333
encontrado	gato	macho	313
encontrado	gato	não sei	313
perdido	cachorro	fêmea	525
perdido	cachorro	macho	537
perdido	gato	fêmea	497
perdido	gato	macho	503

Fonte: Próprios autores (2015).

- o tempo total gasto armazenando os resultados das comparações de grau de semelhança maior que 0% no banco de dados (*caching*);
- o tempo total de execução do algoritmo para cada amostra, ou seja, a soma dos tempos anteriores mais operações acessórias (*overhead*);
- os tempos mínimo, médio e máximo de comparação de um anúncio com os outros no SGBD, considerando que o tempo total de comparação no SGBD é mais significativo do que os outros tempos totais;
- a quantidade de comparações efetuadas pela implementação em Java;
- a quantidade de resultados armazenados na tabela *cache* do banco de dados.

Por fim, com base nas estatísticas, foram gerados os seguintes gráficos:

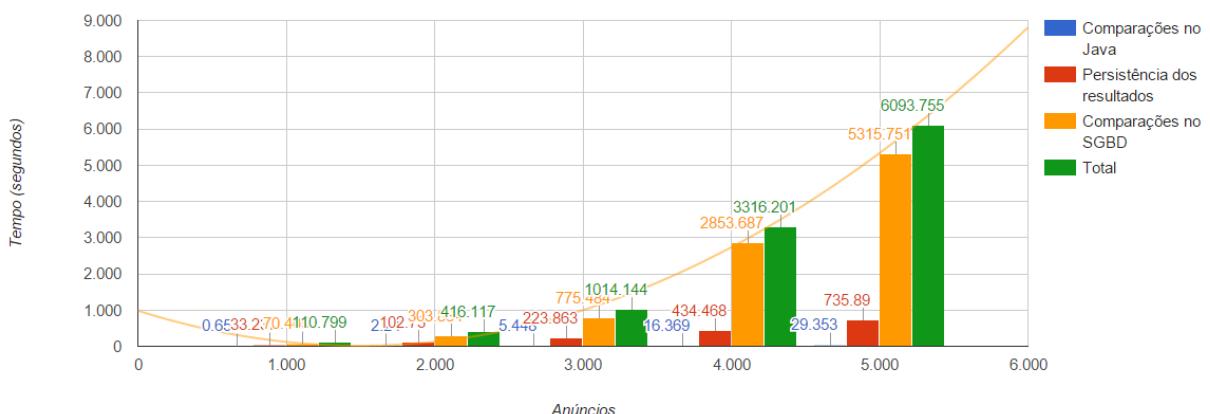
Tabela 18 – Distribuição da amostra do quinto teste.

Natureza	Espécie	Gênero	Quantidade
encontrado	cachorro	fêmea	423
encontrado	cachorro	macho	402
encontrado	cachorro	não sei	424
encontrado	gato	fêmea	380
encontrado	gato	macho	446
encontrado	gato	não sei	417
perdido	cachorro	fêmea	649
perdido	cachorro	macho	600
perdido	gato	fêmea	647
perdido	gato	macho	612

Fonte: Próprios autores (2015).

- tempo de execução de cada fase do algoritmo e total, com a quantidade de anúncios inseridos no eixo x e o tempo em segundos no eixo y , mostrado na Figura 28;
- tempos mínimo, médio e máximo de comparação de um anúncio com os demais feito pelo SGBD, com a quantidade de anúncios inseridos no eixo x e o tempo em milissegundos no eixo y , mostrado na Figura 29;
- tempo de execução de cada comparação feita pelo SGBD, com a ordem do anúncio sendo comparado no eixo x e o tempo em milissegundos no eixo y , para mil, dois mil, três mil, quatro mil e cinco mil anúncios, mostrados nas Figuras 30, 31, 32, 33 e 34, respectivamente.

Figura 28 – Gráfico de tempo total de execução do algoritmo por fase e geral.



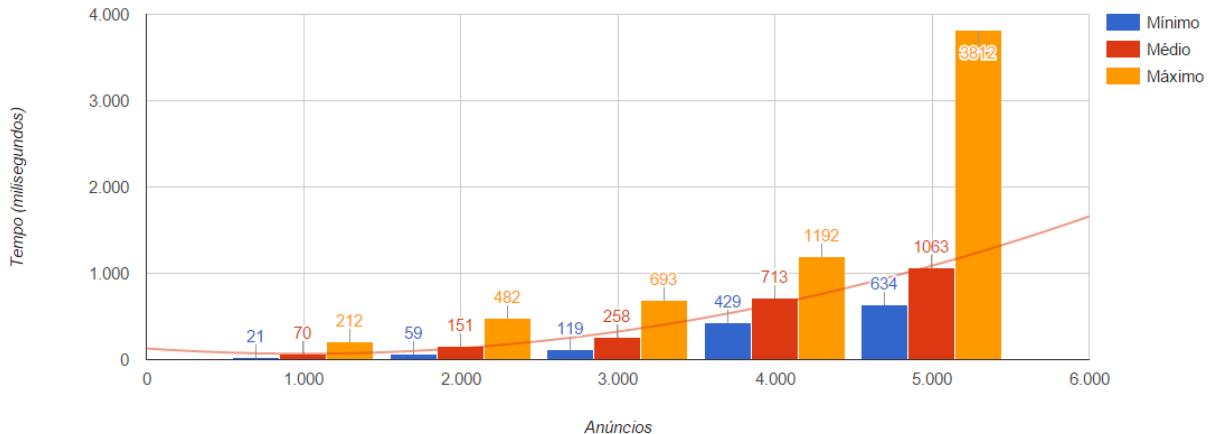
Fonte: Próprios autores (2015).

Tabela 19 – Estatísticas coletadas nos testes.

Estatística	Unidade	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Tempo de comparação total no SGBD	segundos	70,405	303,884	775,484	2.853,687	5.315,751
Tempo de comparação total no Java	segundos	0,653	2,24	5,448	16,369	29,353
Tempo persistindo resultados	segundos	33,237	102,730	223,863	434,468	735,89
Tempo total do algoritmo	segundos	110,799	416,117	1.014,144	3.316,201	6.093,755
Tempo mínimo de comparação no SGBD	milissegundos	21	59	119	429	634
Tempo médio de cada comparação no SGBD	milissegundos	70	151	258	713	1063
Tempo máximo de comparação no SGBD	milissegundos	212	482	693	1192	3812
Comparações no Java	quantidade	121.433	482.617	1.085.902	1.929.261	3.033.724
Persistências de resultados	quantidade	100.475	403.589	912.429	1.609.686	2.514.470

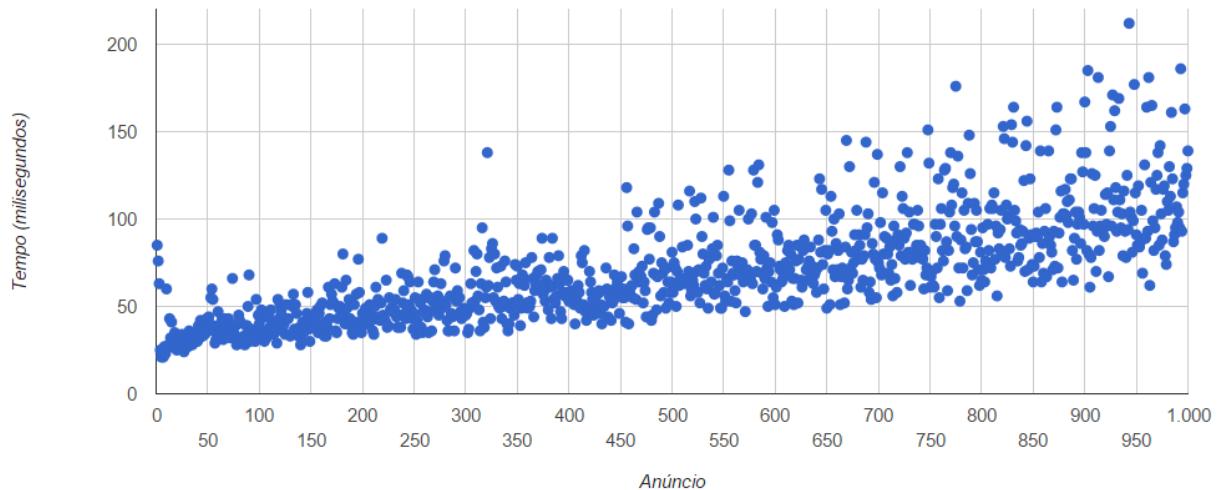
Fonte: Próprios autores (2015).

Figura 29 – Gráfico de tempo de execução do algoritmo por comparação no SGBD.



Fonte: Próprios autores (2015).

Figura 30 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 1000 anúncios.

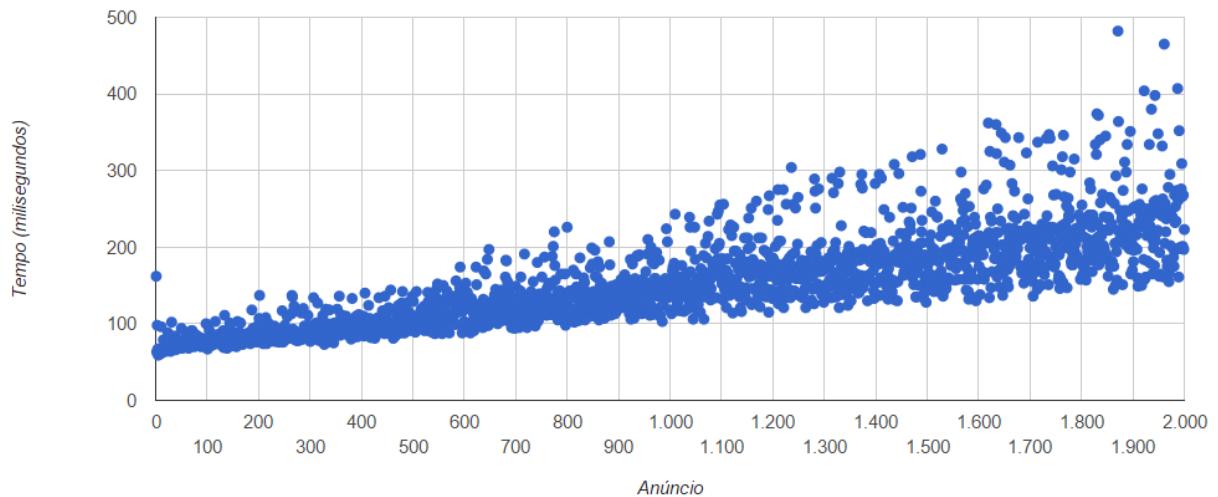


Fonte: Próprios autores (2015).

Com base nas análises feitas, foram feitas as seguintes inferências:

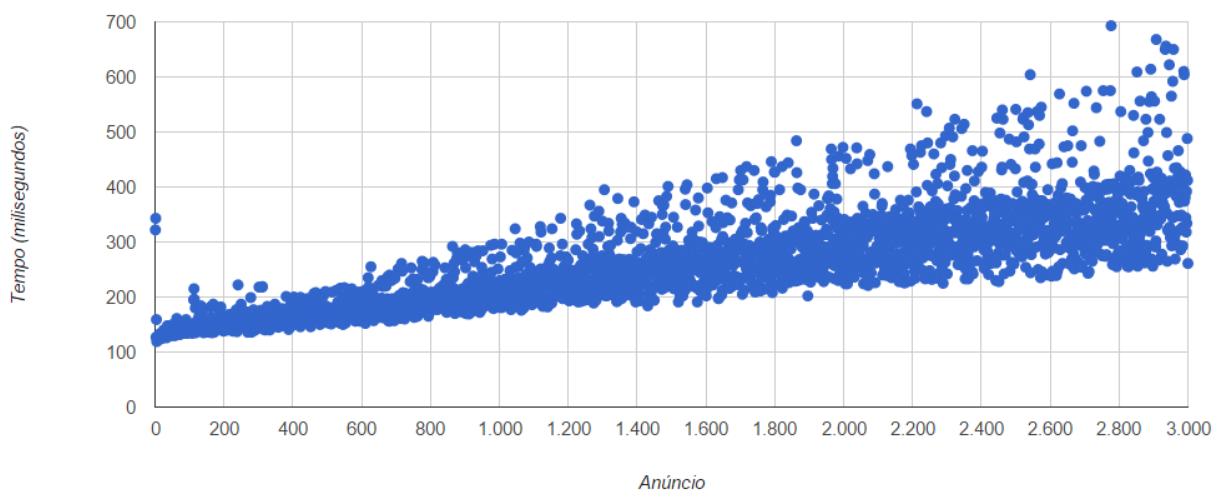
- tanto o tempo total de execução como o tempo médio por comparação no SGBD mostram um crescimento polinomial quadrático ($O(n^2)$), à medida que o número de anúncios cadastrados no sistema cresce, como previsto;
- o tempo de execução do algoritmo no Java é muito menor do que os demais, devido à filtragem feita diretamente no SGBD;
- durante a execução do algoritmo, o crescimento do tempo de cada comparação cresce linearmente como previsto, visto que cada anúncio é comparado com todos os cadastrados antes dele, no pior caso;

Figura 31 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 2000 anúncios.



Fonte: Próprios autores (2015).

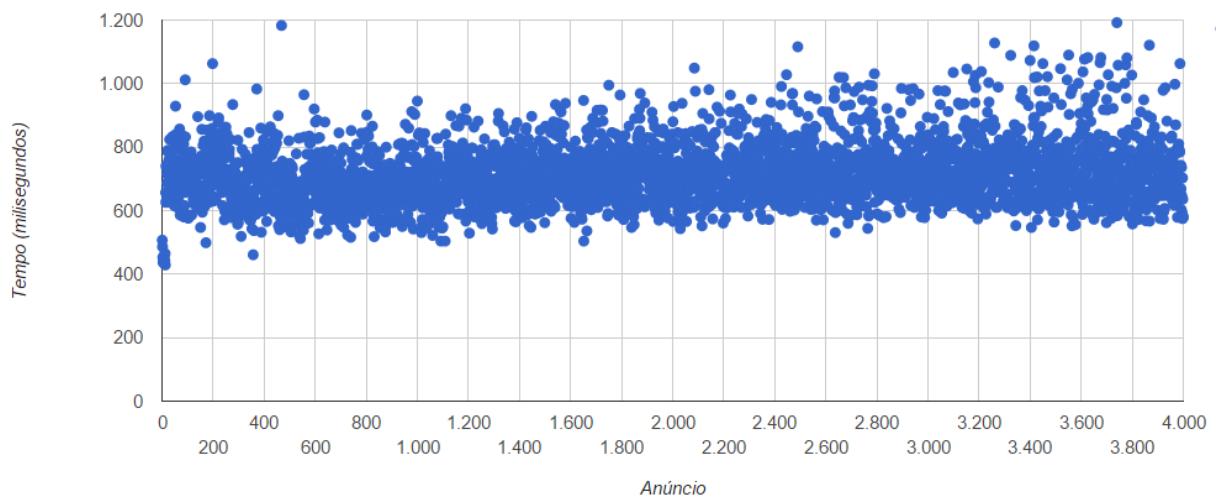
Figura 32 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 3000 anúncios.



Fonte: Próprios autores (2015).

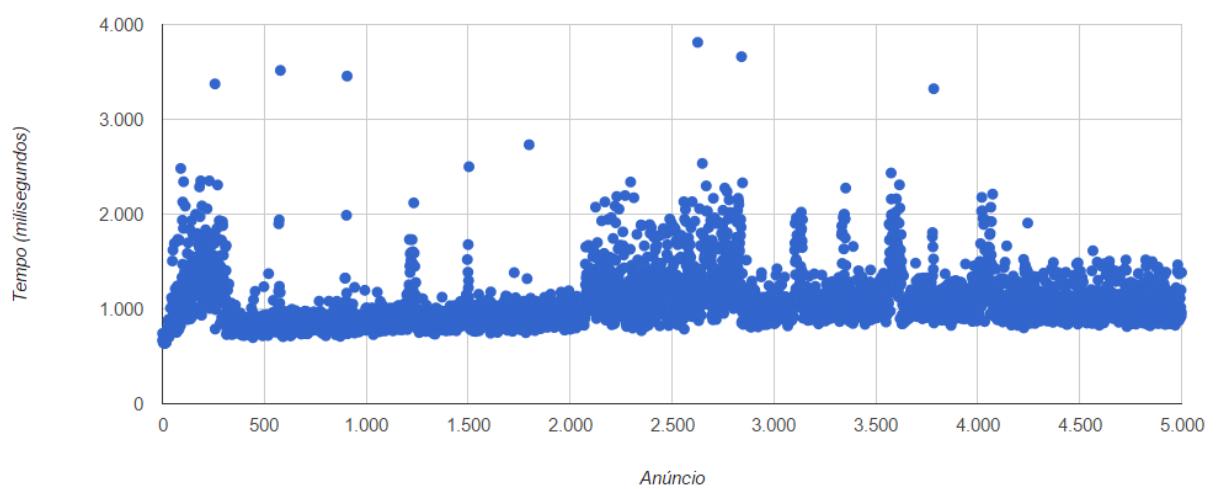
- é possível projetar o tempo de execução do algoritmo para um número maior de anúncios. Para dez mil anúncios, por exemplo, podemos inferir o tempo de execução total do algoritmo a partir do tempo total para cinco mil, utilizando regressão polinomial, pela expressão: $(\sqrt{6093,755s} \div 5000 \times 10000)^2 = 24375,02s = 406,25min.$

Figura 33 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 4000 anúncios.



Fonte: Próprios autores (2015).

Figura 34 – Gráfico de dispersão de tempo de execução por anúncio no SGBD para 5000 anúncios.



Fonte: Próprios autores (2015).

7 CONCLUSÃO

O maior desafio enfrentado na implementação do sistema Bicho Perdido foi o tempo necessário para o aprendizado dos *frameworks* e bibliotecas de *software* utilizadas – como AngularJS, Spring e Hibernate, por exemplo – bem como o uso de tecnologias ainda instáveis ou com especificação incompleta, como diversas APIs do HTML5, em especial *Server-Sent Events* e *Service Workers*. Porém, as dificuldades proporcionaram uma experiência rica em desenvolvimento de sistemas para *web* e móveis.

O aprendizado dos *frameworks* arquiteturais AngularJS e Spring, apesar do tempo gasto, foi convertido em muitos benefícios, tanto na produção de “mais com menos” código, como em sua organização. Além disso, o desenvolvimento das macrocamadas Cliente e Serviço de forma independente possibilitou o desenvolvimento em paralelo de quase todas as funcionalidades do sistema, agilizando a realização do projeto dentro do tempo disponível.

A integração, realizada várias vezes durante a execução do projeto, trouxe diversas dificuldades, principalmente com relação à padronização da comunicação e formatos de dados utilizados entre as macrocamadas, às restrições impostas pelos navegadores da *web* e tecnologias utilizadas, às falhas no *software* reveladas somente após os testes de integração, entre outros problemas. Mas seguiu-se adiante com o trabalho, pensando somente em atingir os objetivos propostos desde o início.

Esta primeira experiência com Sistemas de Recomendação serviu para revelar claramente que as capacidades da Tecnologia da Informação e dos Sistemas de Informação vão muito além de armazenar e recuperar informações. As possibilidades estão cada vez mais próximas das capacidades humanas, mudando cada vez mais rápido o paradigma de interação de pessoas com máquinas, tornando a experiência mais rica.

Apesar de não ter havido tempo suficiente para implementar todo o projeto, o resultado obtido foi mais do que satisfatório. O Bicho Perdido tem total condição de ser disponibilizado ao público em geral e de cumprir seu objetivo primário, que é de devolver os muitos animais de estimação – membros “peludos” da família – que se perdem aos seus lares rapidamente e em segurança.

7.1 TRABALHOS FUTUROS

Considerando-se a premissa de que mesmo o que é bom sempre pode ser melhorado, a seguir estão relacionadas melhorias projetadas para o futuro:

- integração com redes sociais, para ampliar a capacidade de divulgação dos anúncios;
- algoritmo de recomendação distribuído, para melhor desempenho;
- compatibilidade com mais navegadores *web* e sistemas operacionais móveis;
- criação de aplicações móveis híbridas ou nativas, para ampliação das capacidades do sistema;
- suporte ao envio e armazenamento de vídeos, sem dependência de serviços de terceiros;
- histórico das notificações (“caixa de mensagens”);
- otimização do algoritmo com o uso de bases de dados *NoSQL*.

REFERÊNCIAS

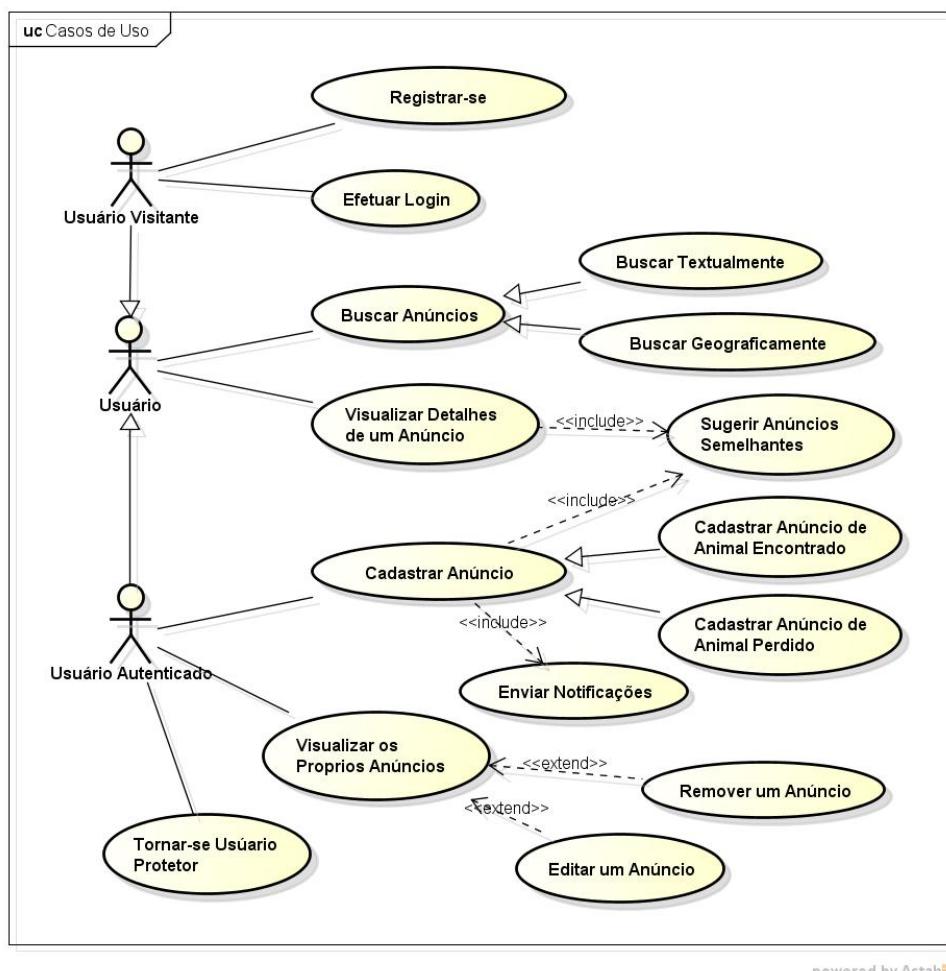
- AGENDAPET. *Animal perdido: saiba os recursos que te ajudam a recuperar seu amigo.* Bolsa de Mulher, 2015. Disponível em: <<http://www.bolsademulher.com/pet/perdeu-seu-pet-saiba-o-que-fazer-e-os-recursos-disponiveis-para-trazer-seu-amigo-de-volta>>. Acesso em: abr. 2015.
- ASSOCIAÇÃO BRASILEIRA DA INDÚSTRIA DE PRODUTOS PARA ANIMAIS DE ESTIMAÇÃO. *Abinpet divulga dados consolidados do mercado pet referentes a 2013.* Disponível em: <<http://abinpet.org.br/imprensa/noticias/abinpet-divulgadados-mercado-pet-2013/>>. Acesso em: abr. 2015.
- BRAY, T. The javascript object notation (json) data interchange format. 2014.
- CHAPPELL, D. A.; JEWELL, T. *Java web services.* [S.l.]: Tecniche Nuove, 2002.
- CONSELHO FEDERAL DE MEDICINA VETERINÁRIA. *População de gatos cresce em média 8% ao ano no Brasil e está perto de se igualar a de cães.* 2014. Disponível em: <<http://portal.cfmv.gov.br/portal/noticia/index/id/4064/secao/6>>. Acesso em: abr. 2015.
- COSTA, E. C. et al. Aspectos psicossociais da convivência de idosas com animais de estimação: uma interação social alternativa. *Psicologia: teoria e prática*, Universidade Presbiteriana Mackenzie, v. 11, n. 3, p. 2–15, 2009.
- ENCONTRA-ME.ORG. *Estatísticas de Animais Desaparecidos.* Disponível em: <<https://www.encontra-me.org/desaparecidos/estatisticas>>. Acesso em: abr. 2015.
- EVANGELISTA-EPPENSTEIN, T. *How many pets were lost, ran away or stolen by the time you finished reading this article?* 2013. Disponível em: <<http://blogcenter.readingeagle.com/a-closer-look-at-animal-welfare-issues/2013/03/18/how-many-pets-were-lost-ran-away-or-stolen-by-the-time-you-finished-reading-this-article/>>. Acesso em: abr. 2015.
- FERREIRA, E.; EIS, D. Html 5—curso w3c escritório brasil. *São Paulo: W3C*, 2010.
- FIELDING, R. T. *Architectural styles and the design of network-based software architectures.* Tese (Doutorado) — University of California, Irvine, 2000.
- INTERNET ENGINEERING TASK FORCE. *RFC 2616.* 1999. Disponível em: <<https://tools.ietf.org/html/rfc2616>>. Acesso em: abr. 2015.
- INTERNET ENGINEERING TASK FORCE. *RFC 2818.* 2000. Disponível em: <<https://tools.ietf.org/html/rfc2818>>. Acesso em: nov. 2015.
- JANNACH, D. et al. *Recommender systems: an introduction.* [S.l.]: Cambridge University Press, 2010.
- JAPAN ELECTRONICS AND INFORMATION TECHNOLOGY INDUSTRIES ASSOCIATION. *Exchangeable image file format for digital still cameras: Exif Version 2.2.* 2002.

- KALIN, M. *Java Web Services: Implementando*. Tradução de: R. Marques. Rio de Janeiro, RJ: Alta Books, 2009.
- MOREIRA, J. L. K. *Qual é a Distância entre dois Pontos na Superfície da Terra?* 2002. Disponível em: <<http://staff.on.br/jlkm/geopath/>>. Acesso em: jun. 2015.
- RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: *Recommender systems handbook*. [S.l.]: Springer, 2011. p. 1–35.
- ROBERTSON, A. R. The CIE 1976 color-difference formulae. *Color Research & Application*, Wiley Online Library, v. 2, n. 1, p. 7–11, 1977.
- SUTHERS-MCCABE, H. M. Take one pet and call me in the morning. *Generations*, American Society on Aging, v. 25, n. 2, p. 93–95, 2001.
- WEISSTEIN, E. W. *Great Circle*. 2002. Disponível em: <<http://mathworld.wolfram.com/GreatCircle.html>>. Acesso em: jun. 2015.

APÊNDICE A – MODELAGEM

Neste apêndice, será apresentada a modelagem do Bicho Perdido, em UML¹, feita durante o projeto do sistema e utilizada para guiar o desenvolvimento. A seguir, são apresentados os casos de uso (UCs) do sistema, ilustrados pelo diagrama de casos de uso da Figura 35.

Figura 35 – Diagrama de casos de uso.



Fonte: Próprios autores (2015).

UC01 - Registrar-se Um usuário visitante informa ao sistema seu nome, telefone, *email* e uma senha para acesso posterior e o sistema armazena as informações e retorna uma mensagem de sucesso. A entidade usuário está ilustrado pelo diagrama de classes (DC) da Figura 36 e este UC, pelo diagrama de classes de análise (DCA) da Figura 38.

¹ Unified Modeling Language ou linguagem de modelagem unificada (tradução nossa).

UC02 - Efetuar login O usuário visitante informa ao sistema seu e-mail e senha de acesso, o sistema verifica a correspondência com dados de usuário previamente registrados e sinaliza sucesso. Este UC também está detalhado no DCA da Figura 38.

UC03 - Buscar anúncios textualmente O usuário informa valores de atributos como critérios de busca (filtro) e o sistema retorna uma lista de anúncios correspondentes. Este UC está ilustrado pelo DCA da Figura 39.

UC04 - Buscar anúncios geograficamente O usuário seleciona uma determinada região em um mapa e o sistema indica os locais das ocorrências anunciadas nessa região. Este UC também está detalhado no DCA da Figura 39.

UC05 - Sugerir anúncios semelhantes O sistema recebe um anúncio de entrada e uma natureza alvo, calcula a semelhança desta entrada com os demais anúncios da natureza alvo cadastrados, através de um algoritmo de recomendação e retorna uma lista de anúncios ordenada por maior semelhança.

UC06 - Visualizar detalhes de um anúncio O usuário solicita os detalhes de um anúncio e o sistema apresenta todas as informações do anúncio solicitado. Também apresenta o resultado do UC05, dado como entrada o anúncio solicitado e sua natureza análoga como alvo. Por exemplo: usuário solicita detalhes de um anúncio de animal **perdido** e UC05 é executado com natureza **achado** como alvo. Este UC está ilustrado pelo DCA da Figura 40.

UC07 - Enviar notificações O sistema recebe um anúncio e uma lista de usuários como entrada e notifica (por *e-mail* e *mobile*) esses usuários sobre a existência daquele anúncio.

UC08 - Cadastrar anúncio O usuário fornece as muitas informações relativas a um anúncio de animal perdido ou encontrado – conforme relacionado na seção anterior e detalhado no DC da Figura 37 – o sistema armazena as informações, sinaliza sucesso, inicia o UC05 com o anúncio recém cadastrado e sua natureza como entradas, apresentando seu resultado, bem como o UC07, tendo como entrada o anúncio cadastrado e o resultado do UC05. Este UC está detalhado no DCA da Figura 41.

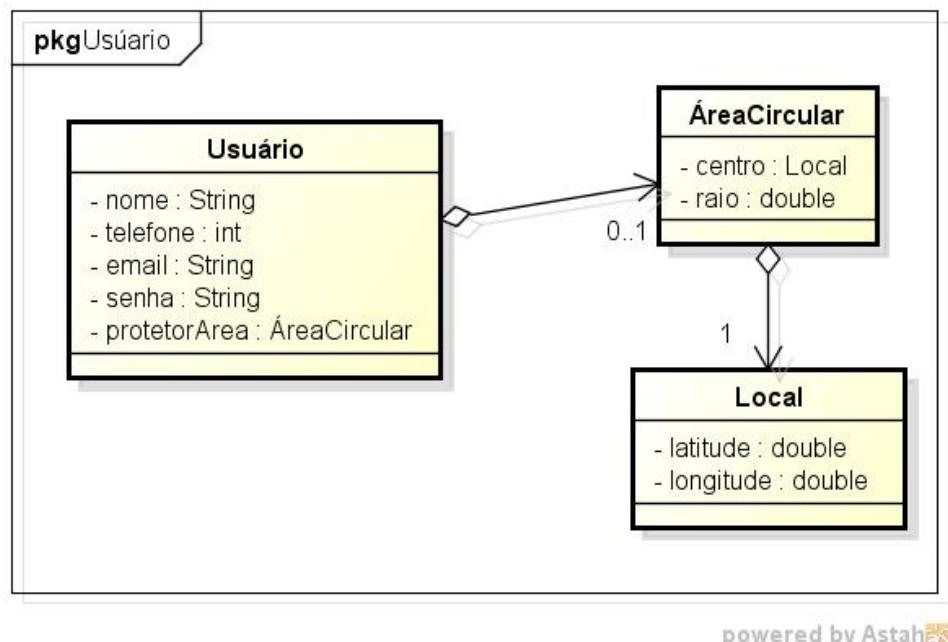
UC09 - Visualizar os próprios anúncios O sistema apresenta ao usuário uma lista de todos os seus anúncios cadastrados. Este UC está ilustrado pelo DCA da Figura 42.

UC10 - Editar um anúncio O usuário informa ao sistema o que deseja alterar em seu anúncio, o sistema armazena as alterações e sinaliza sucesso. Este UC também está detalhado no DCA da Figura 42.

UC11 - Remover um anúncio O usuário indica qual anúncio deseja remover, o sistema remove e sinaliza sucesso. Este UC também está ilustrado pelo DCA da Figura 42.

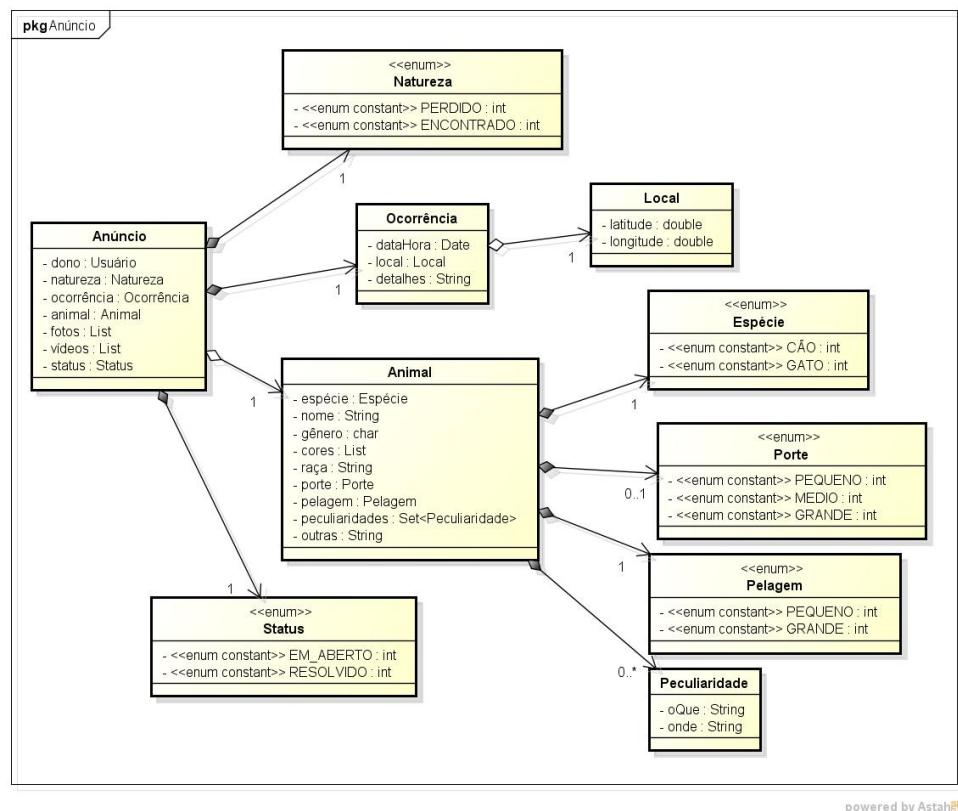
UC12 - Tornar-se usuário protetor O usuário indica uma área circular que deseja monitorar (ser notificado de cada novo anúncio, cujo local da ocorrência esteja dentro da área indicada), o sistema armazena a informação e sinaliza sucesso. Este UC também está detalhado no DCA da Figura 38.

Figura 36 – Diagrama de classes detalhando a entidade Usuário.



Fonte: Próprios autores (2015).

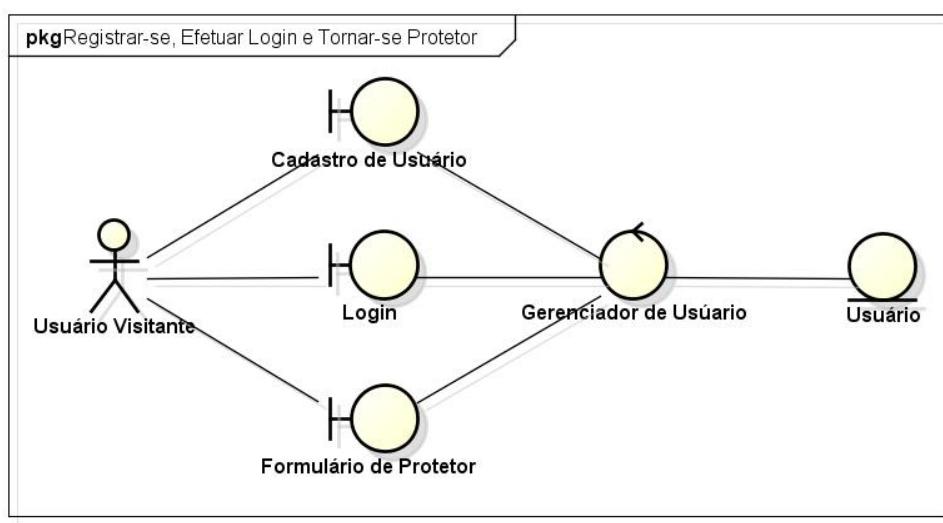
Figura 37 – Diagrama de classes detalhando a entidade Anúncio.



powered by Astah

Fonte: Próprios autores (2015).

Figura 38 – Diagrama de classes de análise de registro, login e protetor.



powered by Astah

Fonte: Próprios autores (2015).

Figura 39 – Diagrama de classes de análise da busca de anúncios.

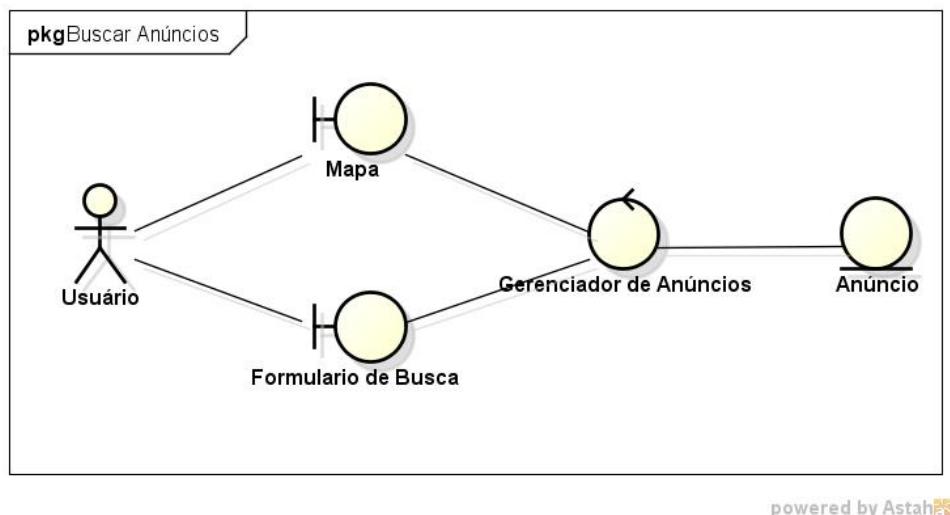


Figura 40 – Diagrama de classes de análise dos detalhes de um anúncio.

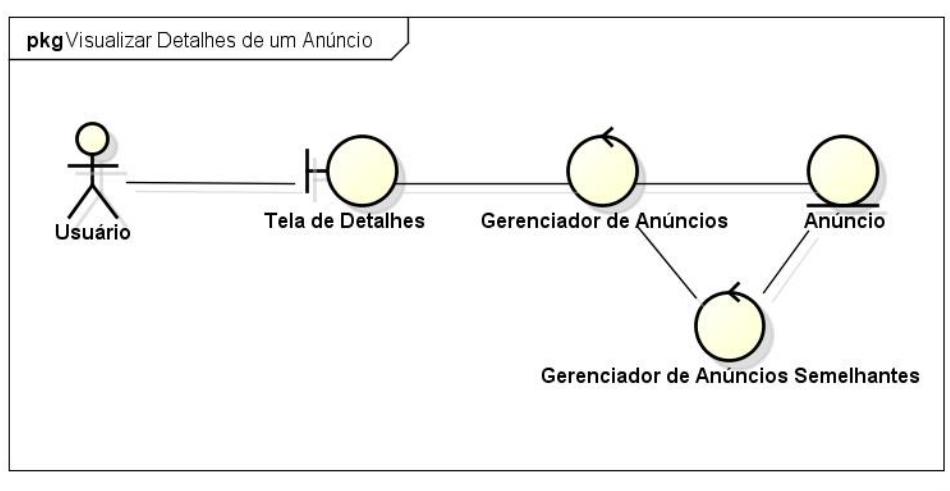
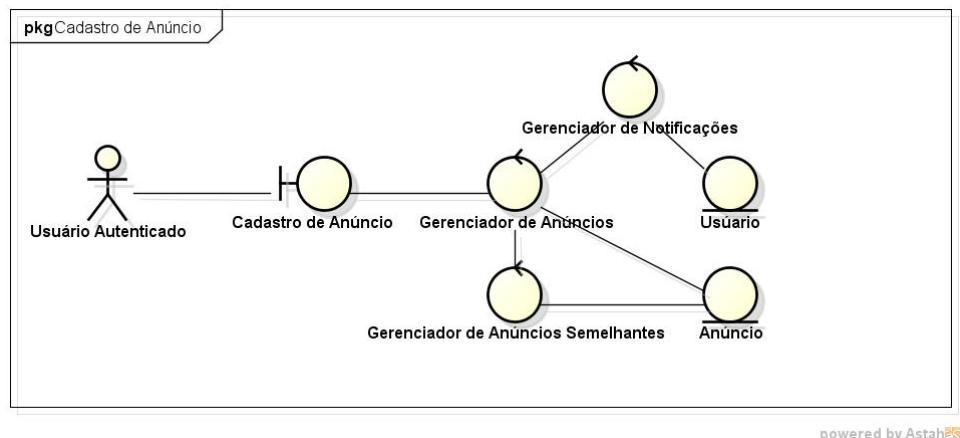


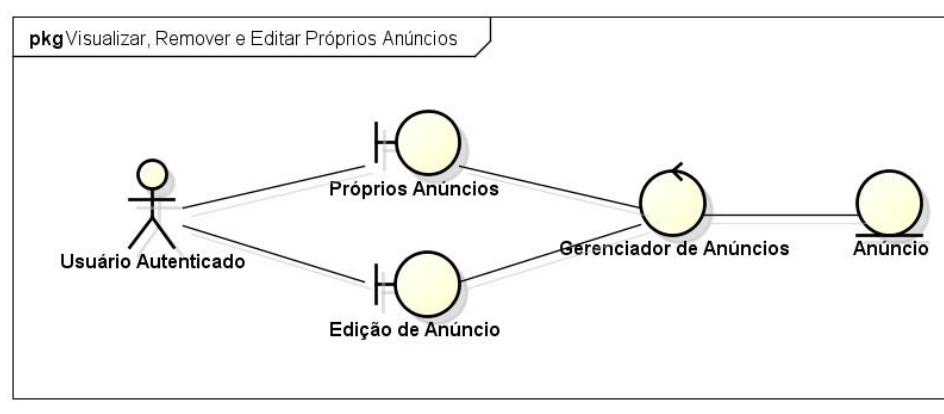
Figura 41 – Diagrama de classes de análise do cadastro de anúncio.



powered by Astah

Fonte: Próprios autores (2015).

Figura 42 – Diagrama de classes de análise dos próprios anúncios.



powered by Astah

Fonte: Próprios autores (2015).

APÊNDICE B – EXEMPLOS DE USO

Neste apêndice, serão apresentados exemplos práticos de uso do sistema Bicho Perdido, passo-a-passo, com imagens ilustrativas.

B.1 CADASTRAR-SE COMO USUÁRIO

Para cadastrar um usuário no sistema, é necessário realizar os passos a seguir, ilustrados pela Figura 43:

1. acessar a tela de cadastro de usuário pela opção “Cadastro” da barra de navegação;
2. preencher os campos “Nome”, “Telefone para contato”, “Endereço de e-mail”, “Senha” e “Confirmação da senha”;
3. clicar ou tocar no botão “Cadastrar-se”.

Figura 43 – Tela de cadastro de usuário.

Fonte: Próprios autores (2015).

B.2 AUTENTICAR-SE E TORNAR-SE PROTETOR

Um usuário cadastrado pode autenticar-se no sistema realizando os passos a seguir, ilustrados pela Figura 44:

1. acessar a tela de autenticação pela opção “Identificar-se” da barra de navegação;

2. preencher os campos “Endereço de e-mail” e “Senha”;
3. clicar ou tocar no botão “Entrar”.

Figura 44 – Tela de autenticação de usuário.



Fonte: Próprios autores (2015).

Após autenticar-se, um usuário pode tornar-se um protetor realizando os passos a seguir, ilustrados pela Figura 45:

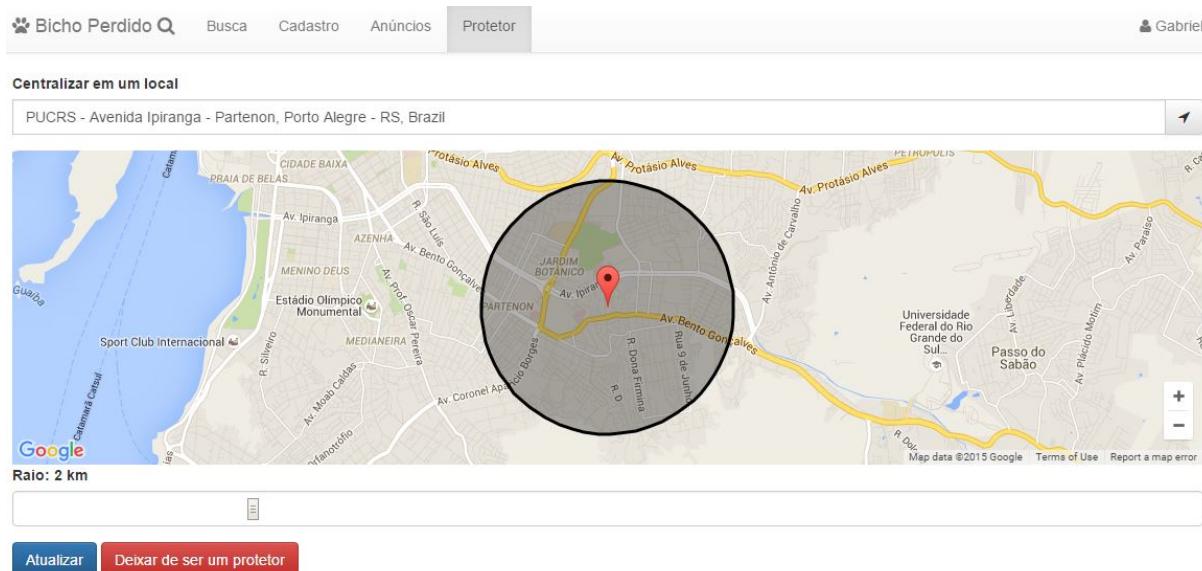
1. acessar a tela de configuração do usuário protetor pela opção “Protetor” da barra de navegação;
2. definir um ponto central no mapa: arrastando o marcador, digitando o endereço no campo “Centralizar em um local” ou solicitando a posição atual do dispositivo clicando ou tocando no botão com a seta;
3. definir o raio em quilômetros da área circular a ser monitorada utilizando o controle deslizante logo abaixo do mapa;
4. clicar ou tocar no botão “Atualizar”.

B.3 CRIAR E GERENCIAR ANÚNCIOS

Usuários autenticados podem criar anúncios de animais perdidos ou encontrados realizando os passos a seguir:

1. acessar a tela de cadastro de anúncio pela opção “Novo anúncio” do menu “Anúncios” da barra de navegação, conforme ilustrado pela Figura 46;
2. clicar ou tocar no botão correspondente à “Natureza do anúncio”, ou seja, se é um “Animal perdido” ou um “Animal encontrado”;

Figura 45 – Tela de configuração do usuário protetor.

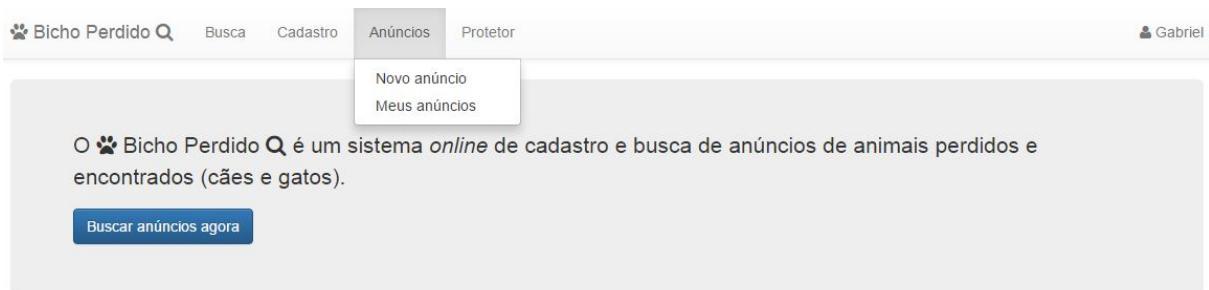


Fonte: Próprios autores (2015).

3. clicar ou tocar no botão “Selecionar...” e selecionar um arquivo de imagem correspondente à “Foto de capa do anúncio” (os itens 2 e 3 estão ilustrados pela Figura 47);
4. preencher “Data e hora da ocorrência” com o momento exato ou aproximado do ocorrido;
5. informar o “Local da ocorrência” arrastando o marcador no mapa, digitando o endereço no campo logo acima ou solicitando a posição atual do dispositivo clicando ou tocando no botão com a seta (os itens 4 e 5 estão ilustrados pela Figura 48);
6. preencher os “Detalhes da ocorrência”, informar “Espécie” e “Gênero” do animal pelos botões correspondentes, preencher o “Nome...” (apenas se houver certeza) e selecionar as “Cores em ordem de predominância”, conforme ilustrado pela Figura 49;
7. preencher a “Raça”, escolher o “Porte” (apenas no caso de estar sendo anunciado um cachorro) e o tamanho da “Pelagem” pelos botões correspondentes, definir as “Peculiaridades” selecionando combinações de tipo (“O quê?”) e local (“Onde?”) e preencher “Mais informações” sobre o animal, conforme ilustrado pela Figura 50;
8. finalizar o cadastro do anúncio clicando ou tocando no botão “Anunciar”.

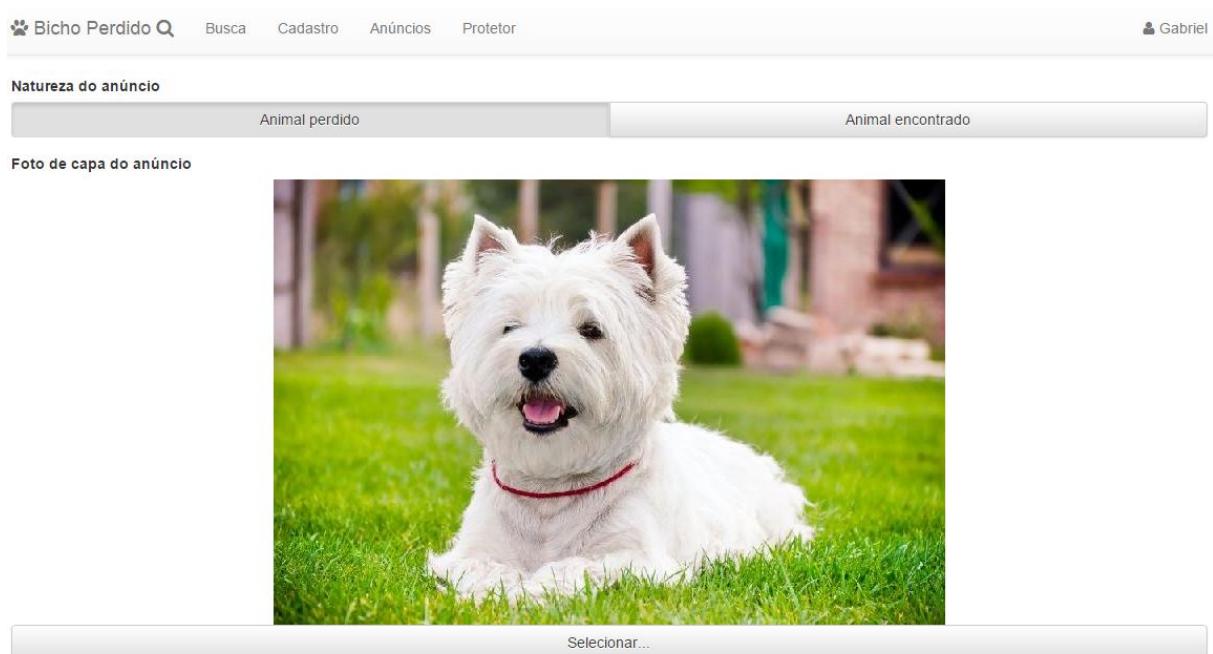
Após o cadastro, o sistema sugere ao usuário anúncios semelhantes ao seu, conforme ilustrado pela Figura 51. Os anúncios cadastrados pelo usuário podem ser visualizados e gerenciados na tela de próprios anúncios, acessada através da opção “Meus anúncios” do menu “Anúncios” e ilustrada pela Figura 52.

Figura 46 – Opções do menu “Anúncios”.



Fonte: Próprios autores (2015).

Figura 47 – Primeira parte da tela de cadastro de anúncio.



Fonte: Próprios autores (2015).

B.4 BUSCAR ANÚNCIOS

Qualquer usuário, autenticado ou não, pode buscar por anúncios “em aberto” (ou seja, casos ainda não marcados como resolvidos pelos anunciantes) no sistema, selecionando a opção “Busca” da barra de navegação. As buscas podem ser realizadas interagindo com um mapa ou informando uma ou mais palavras-chave. Os resultados de qualquer das formas de busca podem ser filtrados por: natureza do anúncio, período – data inicial (“a partir de”) e final (“até”) da ocorrência – espécie e gênero do animal, conforme ilustrado pela Figura 53.

O mapa na tela de busca, acessado pela aba “Mapa”, exibe ícones que apontam para o local de cada ocorrência anunciada. É possível acessar os detalhes do anúncio

Figura 48 – Segunda parte da tela de cadastro de anúncio.

Data e hora da ocorrência
06/12/2015 17:36

Local da ocorrência

PUCRS - Avenida Ipiranga - Partenon, Porto Alegre - RS, Brazil

Fonte: Próprios autores (2015).

Figura 49 – Terceira parte da tela de cadastro de anúncio.

Detalhes da ocorrência ("perdeu-se na rua", "fugiu de casa", etc.)
A coleira rasgou durante o passeio

Espécie
Cachorro Gato

Gênero
Macho Fêmea

Nome do cachorro
Bidu

Cores em ordem de predominância (até 3)
1ª cor Remover
2ª cor Remover

Fonte: Próprios autores (2015).

correspondente clicando ou tocando no ícone, conforme ilustrado pela Figura 54. O significado dos ícones encontram-se no painel “Legenda do mapa”, ilustrado pela Figura 55.

Para buscar por palavras-chave, o usuário deve acessar a aba “Lista”, informar uma ou mais palavras-chave e clicar ou tocar no botão “Buscar”. Após realizar a busca, é exibida uma lista com os anúncios que correspondentes, conforme ilustrado pela Figura 55.

Figura 50 – Quarta parte da tela de cadastro de anúncio.

Bicho Perdido Busca Cadastro Anúncios Protetor Gabriel

2^a cor Remover

Mais cores...

Raça
Bichon Frisé

Porte
Pequeno Médio Grande

Pelagem
Curta Longa

Peculiaridades
1^a peculiaridade Remover

Mancha Cauda

Mais peculiaridades...

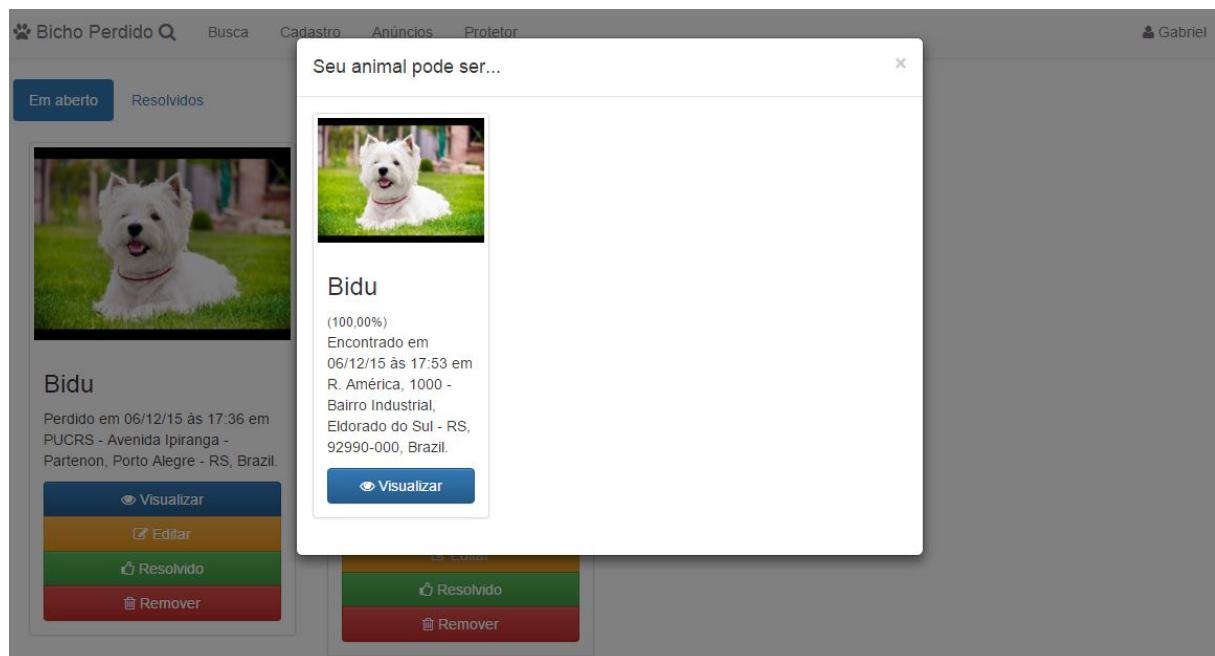
Mais informações (comportamentos, adereços, etc.)

Esta com a coleira, é brincalhão e gosta de arroz

Anunciar

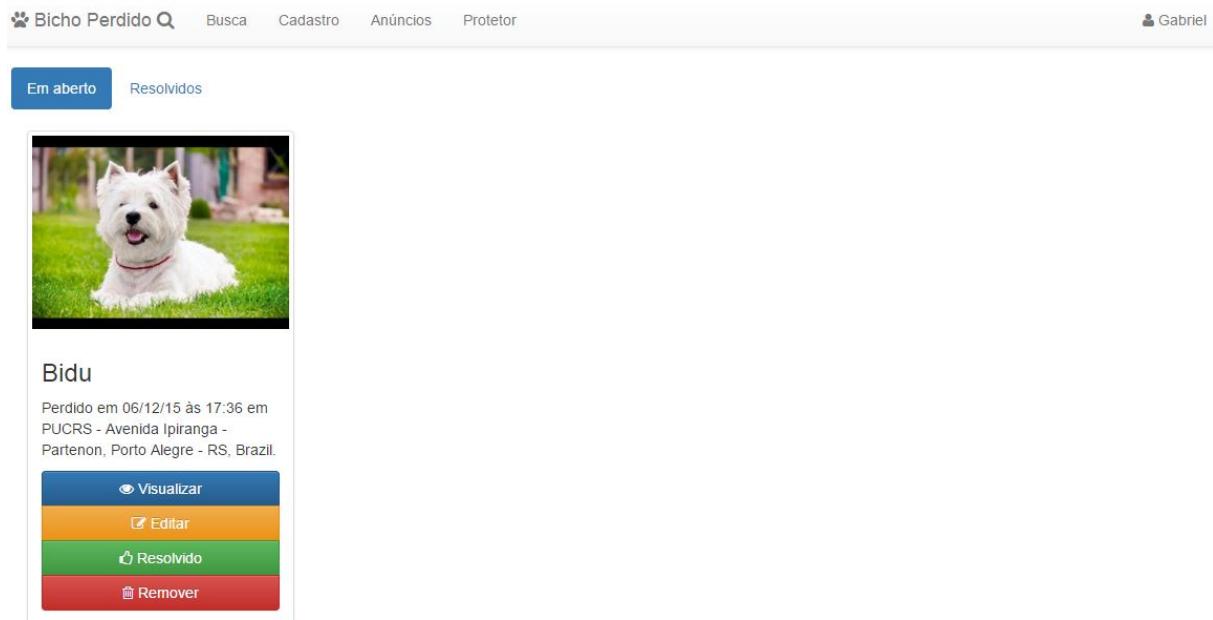
Fonte: Próprios autores (2015).

Figura 51 – Tela de sugestão de anúncios semelhantes após o cadastro.



Fonte: Próprios autores (2015).

Figura 52 – Tela de visualização e gerenciamento dos próprios anúncios.



Fonte: Próprios autores (2015).

Figura 53 – Filtros em comum na tela de busca.

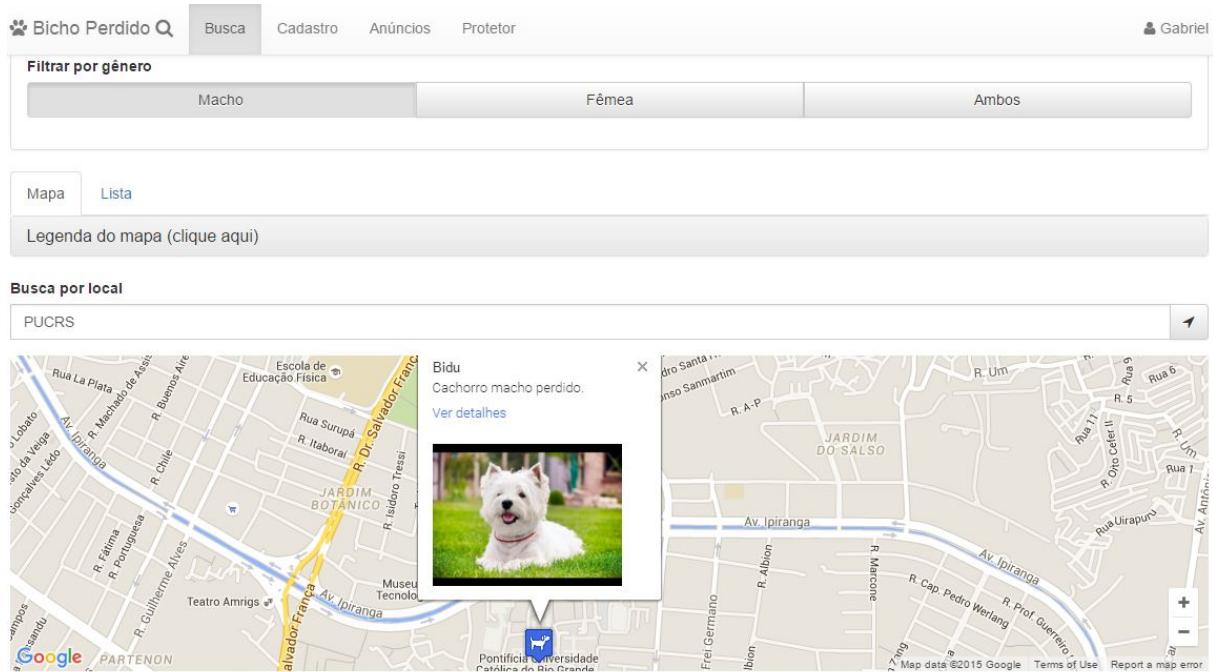


Fonte: Próprios autores (2015).

B.5 EXIBIR OS DETALHES DE UM ANÚNCIO E GERAR CARTAZES

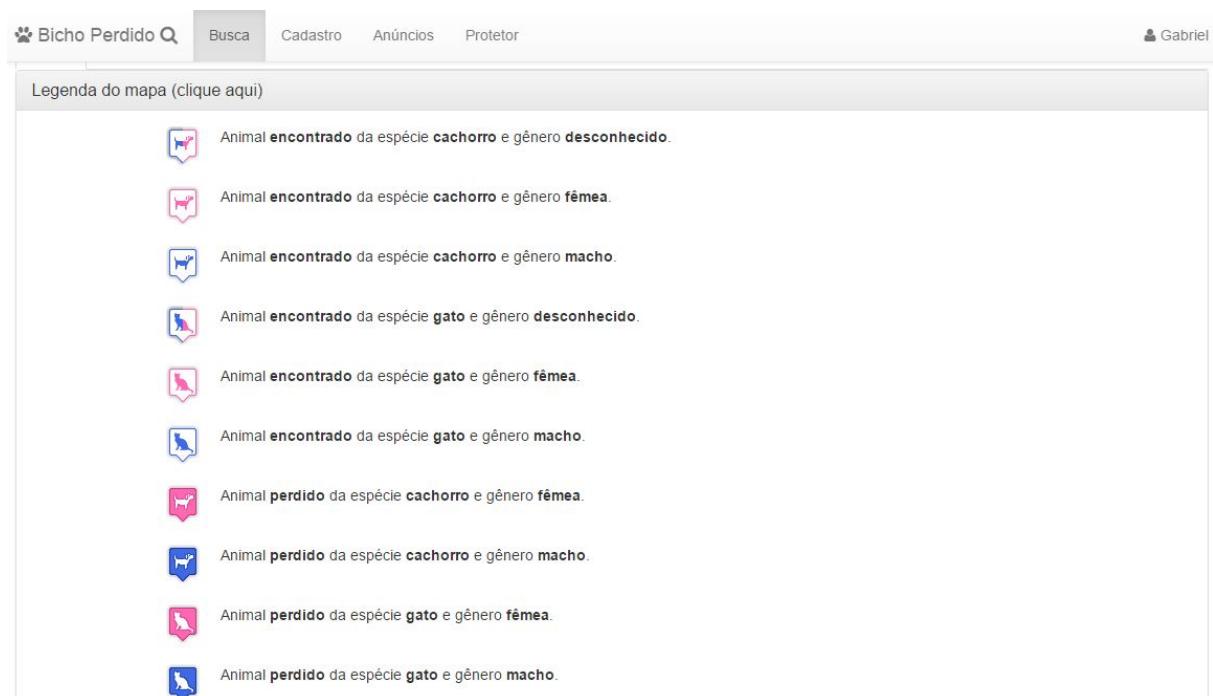
A partir dos resultados de uma busca ou da tela de próprios anúncios, é possível acessar os detalhes de um anúncio. A tela de detalhes de um anúncio exibe todas as

Figura 54 – Mapa na tela de busca.



Fonte: Próprios autores (2015).

Figura 55 – Legenda do mapa na tela de busca.



Fonte: Próprios autores (2015).

informações cadastradas, conforme ilustrado pelas Figuras 57 e 58.

A partir da tela de detalhes, é possível gerar automaticamente um cartaz com as informações do anúncio para ser impresso. Clicando ou tocando no botão “Gerar cartaz”,

Figura 56 – Resultados de uma busca por palavras-chave.

The screenshot shows a search interface for lost pets. At the top, there's a navigation bar with icons for 'Bicho Perdido' (Dog Lost), 'Busca' (Search), 'Cadastro' (Registration), 'Anúncios' (Announcements), 'Protetor' (Protector), and a user profile for 'Gabriel'. Below the search bar, there are two tabs: 'Mapa' (Map) and 'Lista' (List). A search filter section titled 'Filtrar por palavras-chave' (Filter by keywords) contains the terms 'Porto' and 'Alegre' with an 'X' icon, and a placeholder 'Palavra-chave' (Keyword). Below this, a note says 'Ex. Porto Alegre, fugiu, Totó, Poodle, Simpático, etc.' and a 'Buscar' (Search) button. A page navigation bar shows '1' of 1 page. The main content area displays two lost pet entries:

- Bidu**: Cachorro macho perdido em 06/12/15 17:36, PUCRS - Avenida Ipiranga - Partenon, Porto Alegre - RS, Brazil. Includes a photo of a white dog.
- Bethoven**: Cachorro fêmea perdido em 28/11/15 19:12, Av. Benjamin Constant, 220 - São João, Porto Alegre - RS, 90550-003, Brazil. Includes a photo of a black and white dog.

Fonte: Próprios autores (2015).

Figura 57 – Parte superior da tela de detalhes de um anúncio.

This screenshot shows the detailed view of a lost pet announcement for 'Bidu'. At the top, it has the same navigation bar as Figura 56. The title 'Bidu cachorro macho perdido' is displayed above a large photo of a white dog lying on grass. To the right of the photo are several details:

- Data e hora da ocorrência**: 06/12/2015 17:36:00
- Local da ocorrência**: PUCRS - Avenida Ipiranga - Partenon, Porto Alegre - RS, Brazil
- Cores em ordem de predominância**: (A grey bar representing the colors of the dog)
- Raça**: Bichon Frisé
- Porte**: Pequeno
- Pelagem**: Longa
- Contato**: GABRIEL SCHMOELLER, gabriel@bp.com.br, (51) 73737373

Below the main details, there's a 'Galeria' (Gallery) section showing another photo of the dog.

Fonte: Próprios autores (2015).

Figura 58 – Parte inferior da tela de detalhes de um anúncio.

Bicho Perdido Busca Cadastro Anúncios Protetor Gabriel

(51) 73737373

Detalhes da ocorrência
A coleira rasgou durante o passeio

Peculiaridades

O quê?	Onde?
Mancha	Cauda

Mais informações
Esta com a coleira, é brincalhão e gosta de arroz

Mapa da ocorrência

Anúncios semelhantes

Fonte: Próprios autores (2015).

acessa-se a tela de previsão do cartaz, conforme ilustrado pelas Figuras 59 e 60. Após impresso, o cartaz fica como mostrado na Figura 61.

Figura 59 – Parte superior da tela de previsão do cartaz.



Fonte: Próprios autores (2015).

Figura 60 – Parte inferior da tela de previsão do cartaz.

Fonte: Próprios autores (2015).

Figura 61 – Demonstração de um cartaz impresso.

CÃO PERDIDO



“BIDU”

Cão perdido, da raça **Bichon Frisé**, cores “gray” e “white”, porte **pequeno**, pelagem **longa**. Possui **mancha no(a) cauda**.

Perdido em **06/12/15** no(a) **PUCRS - Avenida Ipiranga - Partenon, Porto Alegre - RS, Brazil.**

Informações adicionais:

Esta com a coleira, é brincalhão e gosta de arroz

👤 GABRIEL SCHMOELLER

✉ gabriel@bp.com.br

📞 (51) 73737373

🌐 <http://localhost/bichoperdido/#/anuncio/4>

Fonte: Próprios autores (2015).